

Fire Simulations in Blender 4.5 basics

by [Francesco Milanese](#)

This PDF is free and can be freely shared.

If you enjoy the content, the best way to thank me is to subscribe to my [YouTube channel](#) and give the videos a nice 'Like'. Thank you!

1. [Fundamentals](#)
2. [Principled Volume, Blackbody](#)
3. [Cache, Baking, Offset](#)
4. [Geometry, Inflow, Outflow](#)
5. [Domain Settings, 1/2 \(Gas, Fire, Collections\)](#)
6. [Domain Settings, 2/2 \(Weights, Data, Render, Display\)](#)
7. [Emission, Attributes, Volume Info](#)
8. [Effectors \(Guides, Colliders\)](#)
9. [Force Fields \(Turbulence, Wind, Curve Guide\)](#)
10. [Motion Blur, Filmic, Compositing, Eevee](#)

Fire Simulations in Blender 4.5 basics | 1/10: Fundamentals

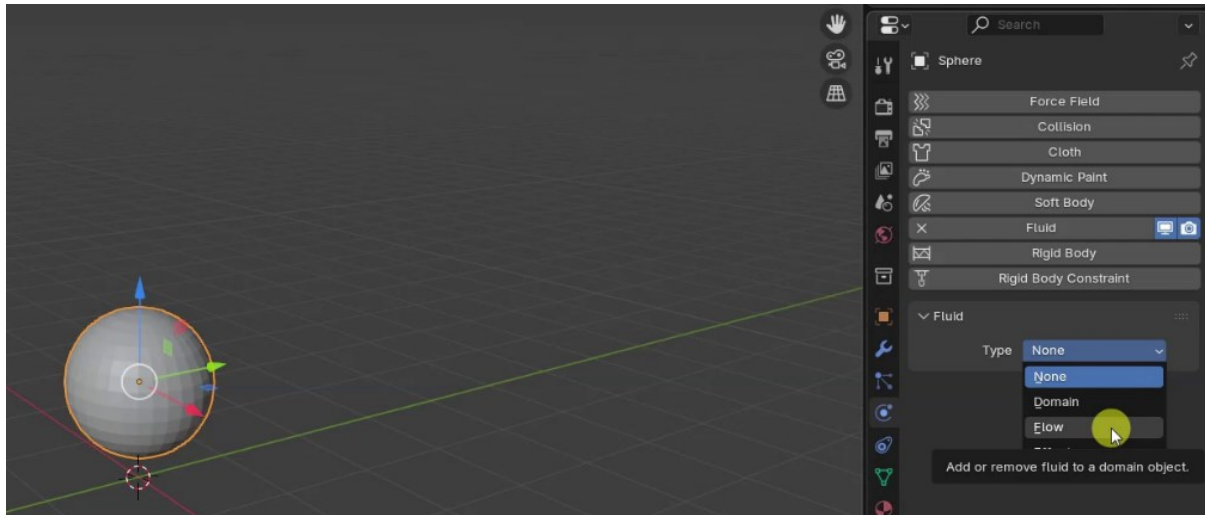
<https://youtu.be/Q-l9WK5JgOA>

Hello everyone! In this first tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll see the two fundamental elements required to create a simulation in Blender.

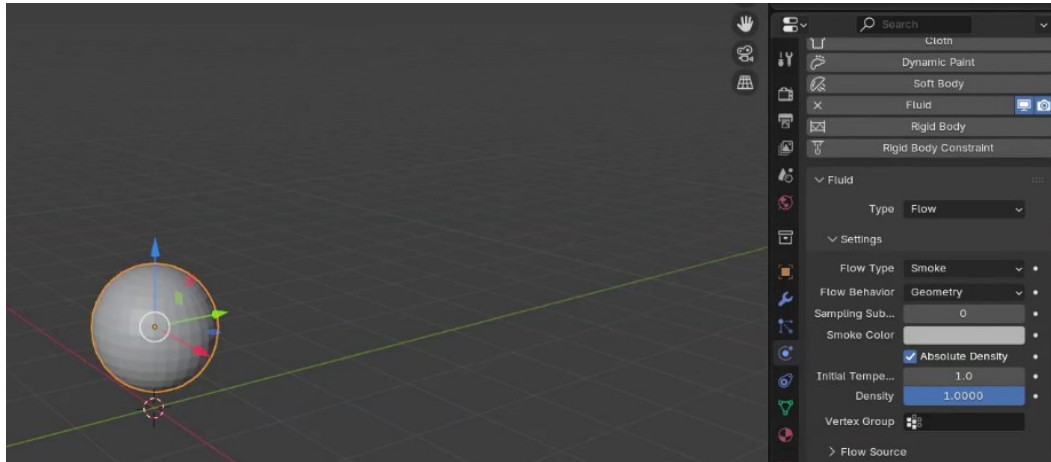
Let's start with an empty Blender scene. To add fire or smoke to the scene, you might think the first step is to insert an object that emits these elements. That assumption is correct, but it represents only half of the fundamental components needed to run a simulation in Blender. Still, let's begin by adding to the scene the object that will act as the emitter.

Blender's fire and smoke simulation tools are located in the Physics tab of each object. The type of component we need to add is Fluid, which handles both liquid and fire or smoke simulations. At first, you'll only see a menu called Type, which doesn't yet distinguish between fluids or other categories, but simply lists the possible actor roles in the simulation.

The tooltips of the various element types already give us some hints about their functions. In particular, we can see that the Flow type is used to add or remove elements from the scene. Since we created the first object to make it an emitter of fire and smoke, we'll choose this type.



Several new options now appear in the component's panel. The first one is Flow Type, which by default is set to Smoke. Opening the menu, we notice that Liquid has only one option, while fire and smoke offer three possible choices: the two elements separately, or their combination. The Fire And Smoke option contains parameters and settings for both, so for the initial part of this tutorial series, I'll select this one.

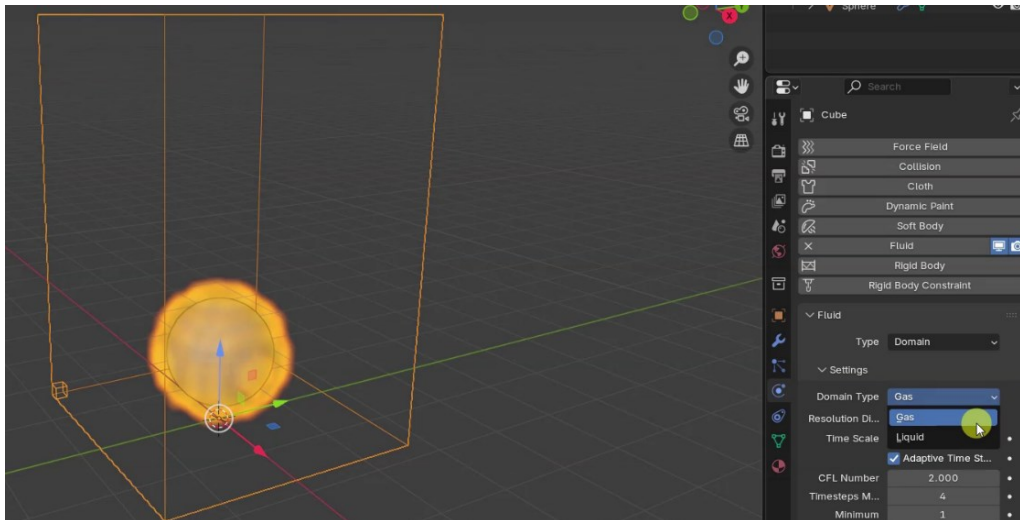


If you're familiar with other types of physics simulations in Blender, such as Cloth or Collision, you might expect that at this point it would be enough to go to the first frame of the Timeline and press the Play button to start the fire and smoke simulation. However, doing this produces no result. So, let's go back to the first frame of the Timeline and introduce the second fundamental actor required for a Fluid simulation: the Domain.

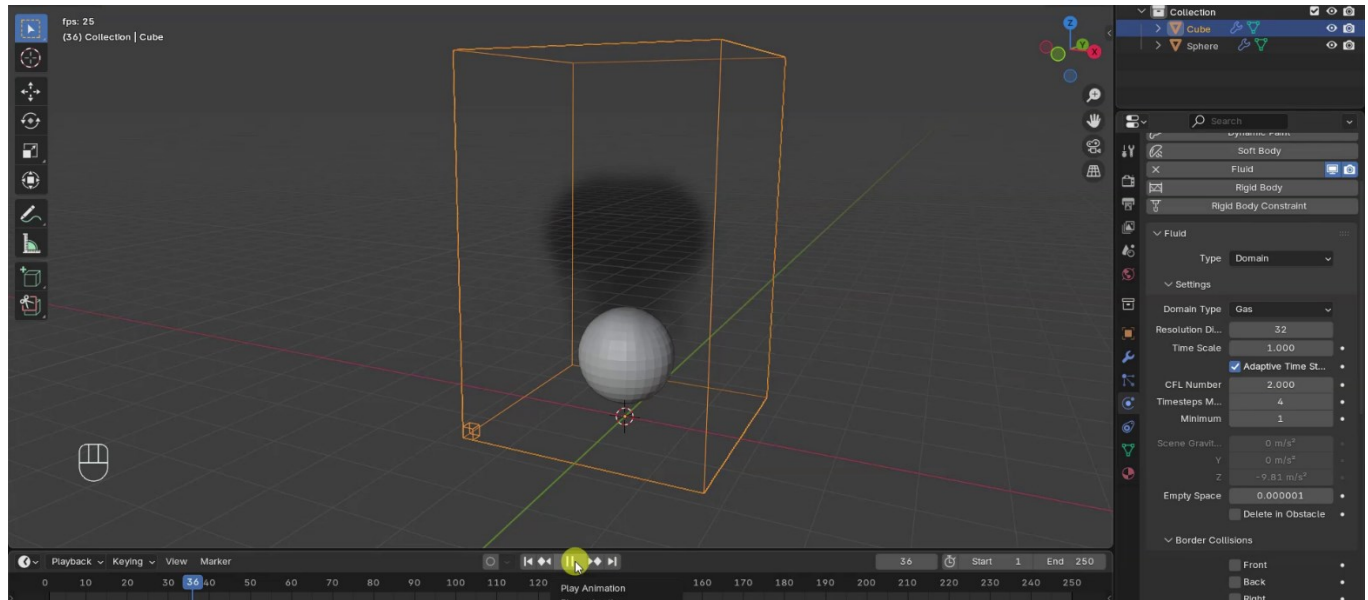
Every Fluid simulation must take place within a 3D space, also called a volume, defined by a mesh. In Blender, this volume is called the Domain of the simulation. This is necessary mainly for performance

reasons, since these simulations require a lot of memory and computation, as we'll see. The Domain can be any shaped object, but Blender will ultimately take its Bounding Box. For this reason, the most obvious choice is a Cube, scaled appropriately.

After enclosing the emitter object inside the Cube that will serve as the Domain, let's select the Cube and add a Fluid component of type Domain, with the subtype set to Gas. The Cube will immediately turn transparent, revealing its structure. You'll also notice a smaller cube appearing in one of the lower corners, and we'll soon see what that's about. The most significant change, however, is that the Inflow object inside the Domain is now surrounded by flames!



With the first frame of the animation set as the current frame in the Timeline, click Play to start the simulation. Something definitely happens, but as a first fire and smoke simulation it seems rather disappointing, since the fire only appears in the first few frames and then only the smoke remains.



The reason for this behavior lies in one of the emitter object's settings, so let's select it. The Fluid component parameter we're interested in is Flow Behavior, which by default is set to Geometry. The tooltip tells us that Geometry uses the current geometry to produce fire, while the Inflow mode

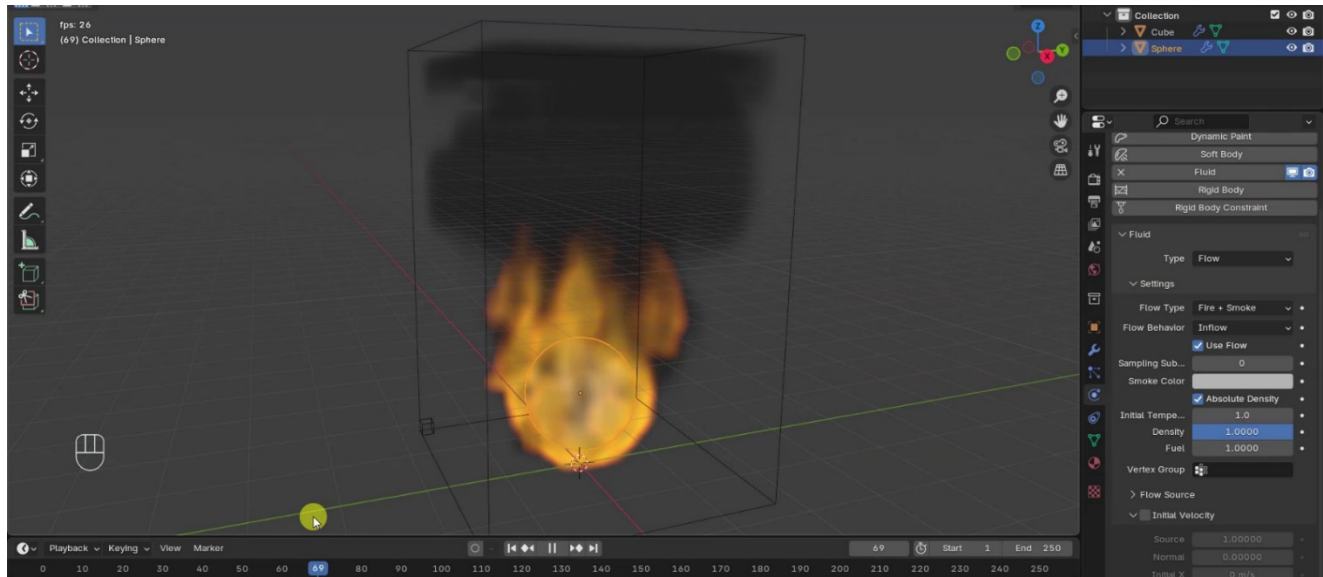
continuously adds fire to the scene. So let's switch the mode to Inflow, go back to the first frame of the Timeline, and click Play again.

The expression “adds fire” for the Inflow is the key to everything, because Inflow objects continuously generate new simulation elements at each frame, starting from the geometry. The Geometry mode, on the other hand, considers the geometry only at the first frame, immediately consuming it, so to speak. In a way, using the Geometry mode is like burning a small piece of paper or having a smoke machine emit a single puff, while Inflow is like a stove burner or a smoke machine that keeps producing smoke continuously.

You may have noticed that the flames, especially at the bottom of the object, look a bit too blocky, or rather cubical, since we're in 3D. The reason for this appearance lies in a Domain parameter called Resolution, which determines how the Domain's volume is subdivided into small cubes.

This resolution is visually represented by the little cube that appears in a lower corner of the Domain.

In fact, especially with low Resolution values, you can clearly see how flames and smoke are divided into cubes of that size.



Fluid simulations work by subdividing the Domain's volume into small cubes called Voxels. These are the smallest units used to calculate the simulation, so a Voxel is the 3D equivalent of a pixel in 2D images. As with pixels, a higher number of Voxels for a given volume results in better quality, but at the cost of more computational power and disk memory. The best approach is to start with the default Resolution value of 32, then adjust the other parameters until you achieve the desired shape for the fire and smoke. After that, increase the Resolution to test further, and finally raise it to a very high value for the final result.

Speaking of memory usage, let's see where Fluid simulation data is stored, especially since I haven't yet saved the project file. This information is found in the Cache section of the Domain object, and from the path shown there we can see that Blender has created a folder on disk to store the simulation files. Since I haven't saved the current project file, this folder is created in a temporary location. So remember not to do what I did. Always save your Blender project file before creating the Domain and other Fluid simulation elements. That way, the cache folder will be created in the same directory as your project file, and you won't have to go searching for it on your disk!

If at this point we switch the viewport shading mode or run a render at an intermediate frame of the Timeline, we won't see either fire or smoke. That's because what we currently see in the 3D Viewport in Solid mode is just the Voxel visualization of fire and smoke in 3D space, but we haven't yet defined their Materials for rendering. We'll cover this in the next episode!

Fire Simulations in Blender 4.5 basics | 2/10: Principled Volume, Blackbody

<https://youtu.be/48inEqSQSNU>

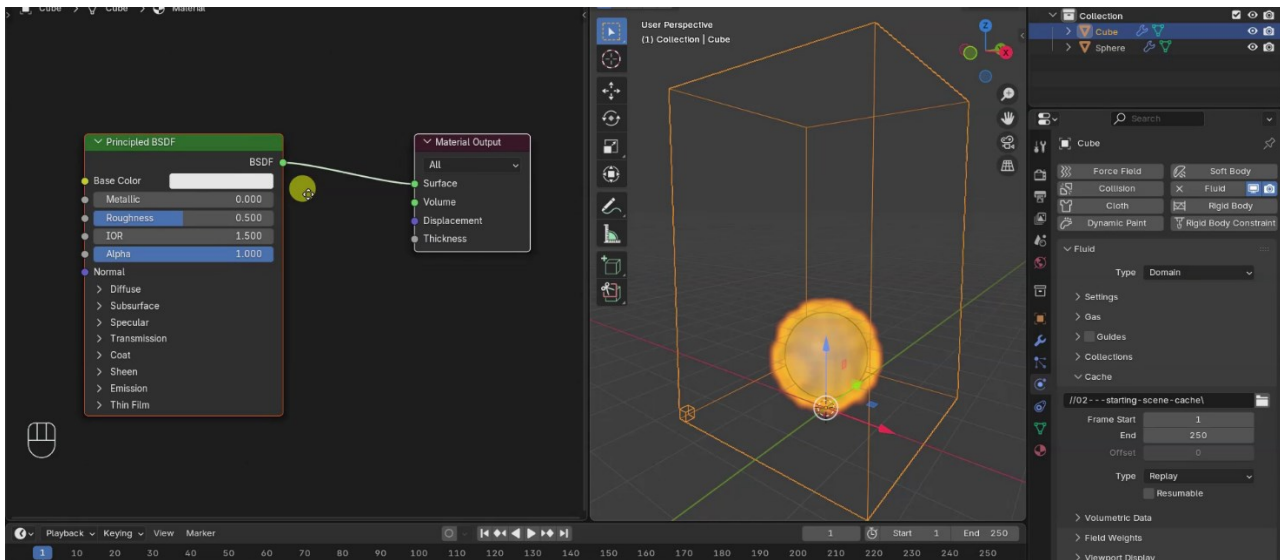
Hello everyone! In this second tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll see how to set up the basic Material in order to render fire and smoke in Cycles.

We'll examine the Principled Volume Shader and the two modes, Emission and Blackbody, that this material provides.

I'm picking up from the scene used at the end of the previous tutorial, but this time I've saved the Blender file and set up a cache folder inside the same directory where the project file is located.

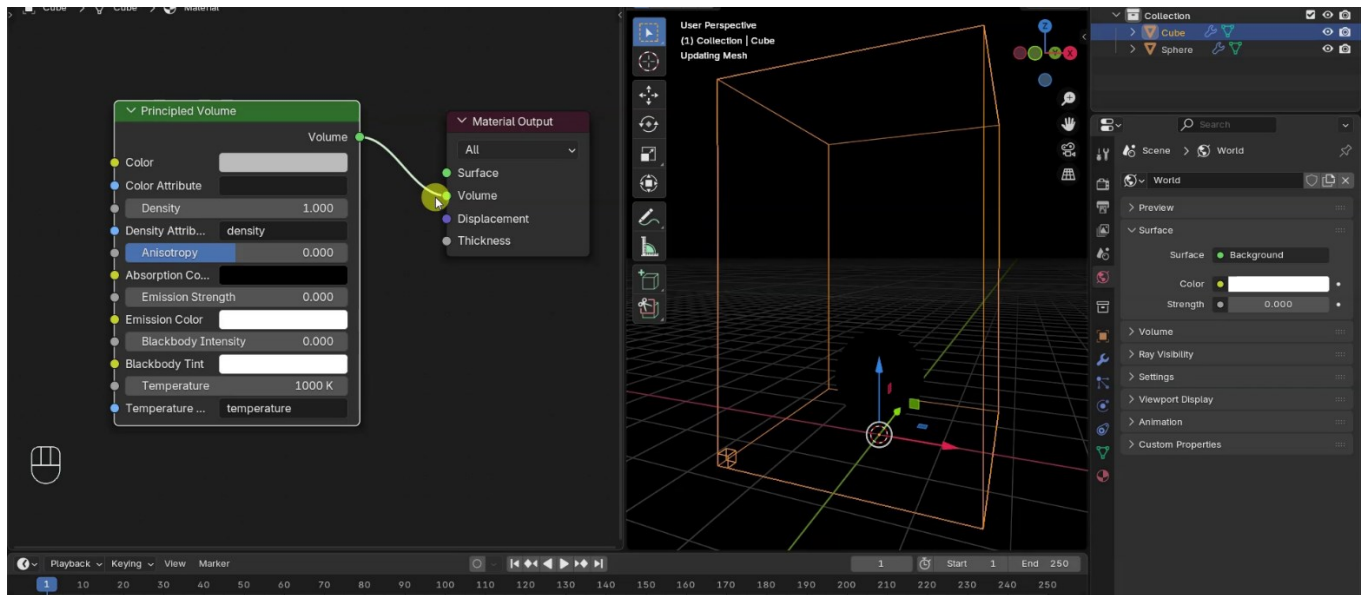
As for the Material for fire and smoke, it should not be defined for the Flow object or objects in the scene, but for the Domain.

So I select the Domain, add a Shading editor to the interface, switch to Rendered view mode, and create a basic Material.

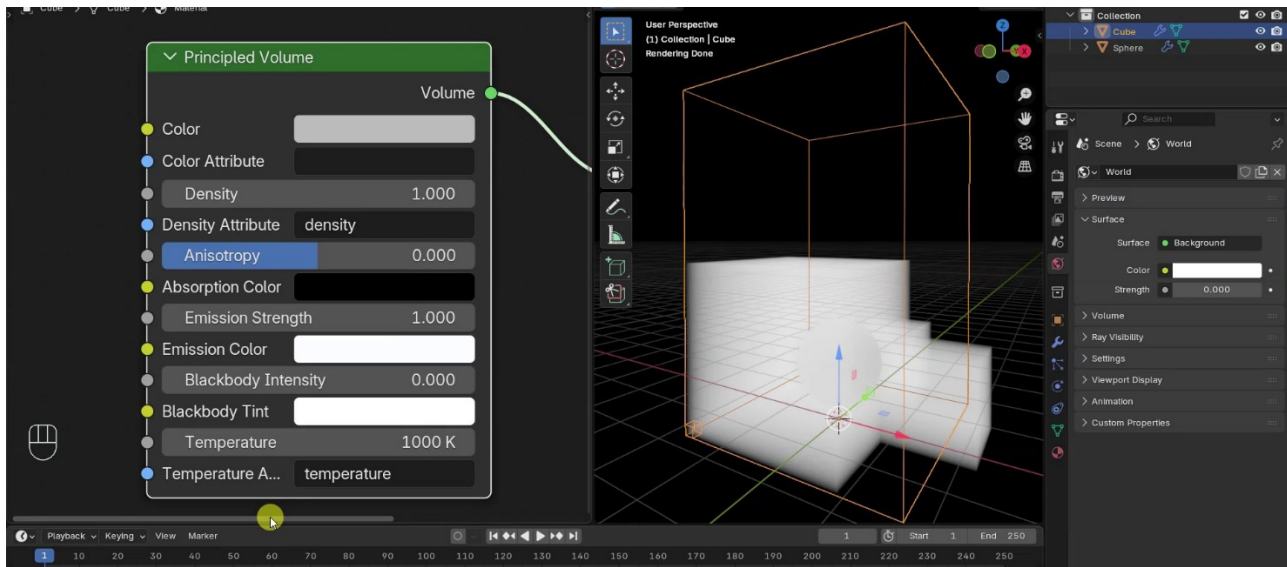


I set the Strength of the Background to 0 in the World tab to make the flames stand out. Playing the animation, we won't see anything yet, because the Principled Shader created with the default material is not suitable for a volumetric material.

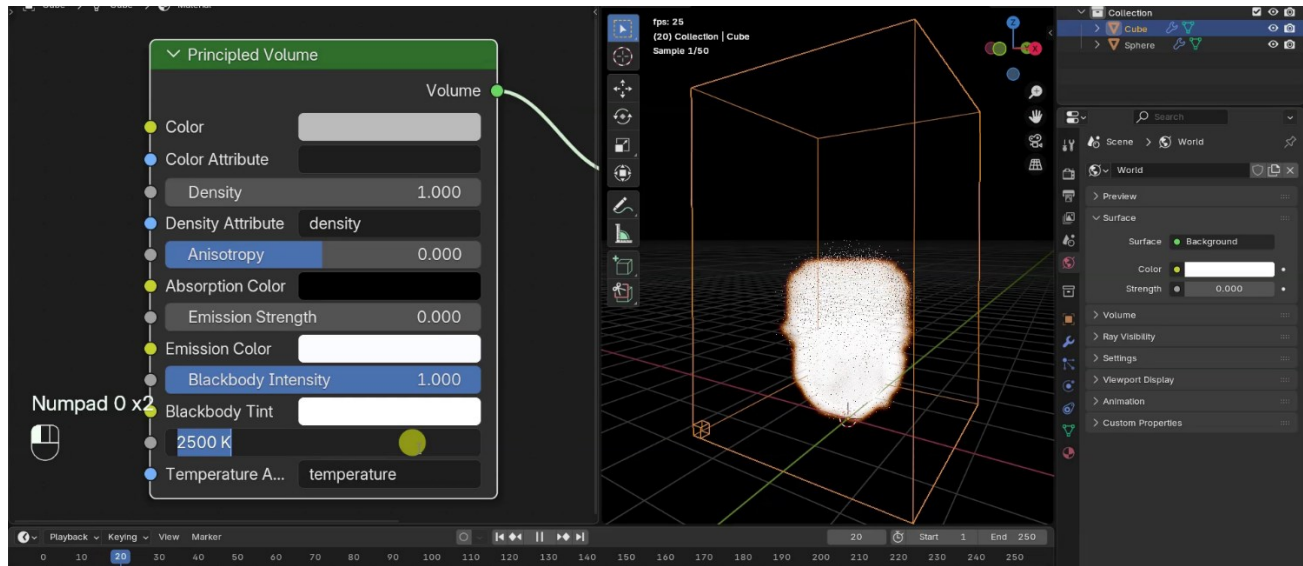
I remove the Principled Shader and insert a Principled Volume Shader, connecting its output to the Volume input of the Material Output node. Even now, when playing the animation, nothing shows up in the Rendered preview. So let's see how to give the flames some color.



Blender provides two different approaches to flame coloring. The first one is called Emission. It is not physically accurate and, at first, its result is quite disappointing, but in reality it gives us great flexibility in managing both the intensity and the colors of the flames. We'll come back to this mode and its uses in much more detail in a later episode.



The second approach is called Blackbody, and this one is physically accurate, especially if the Strength is set to 1 and an appropriate temperature value is chosen for its field. With this tool, we can immediately appreciate a nice flame in the Rendered preview. The Temperature value is expressed in Kelvin, and online you can find many examples of real-world values and their corresponding flame colors.

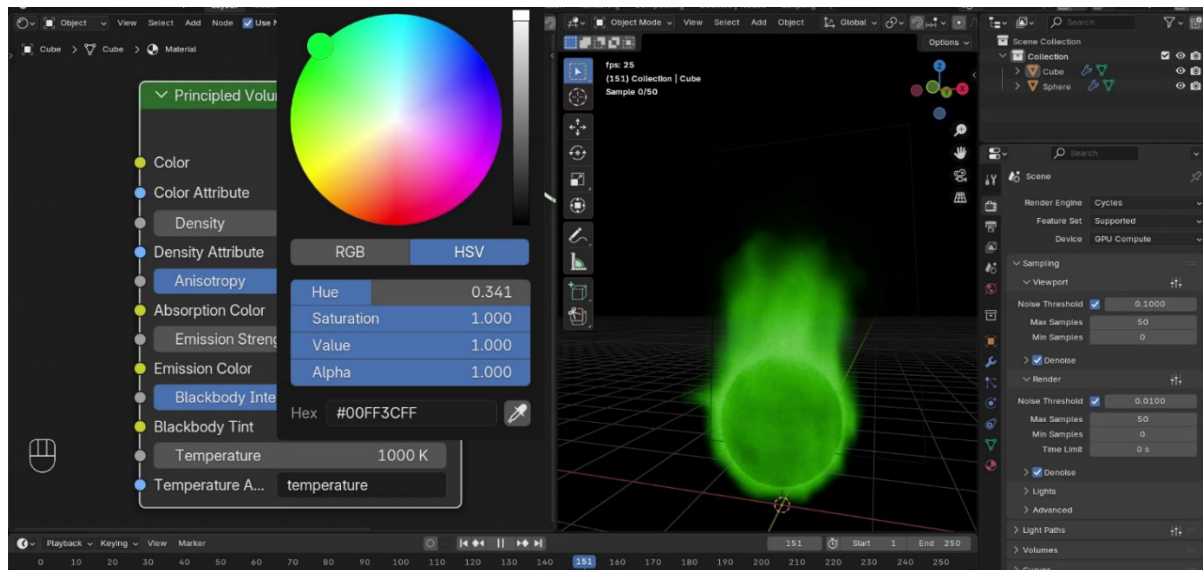


Before continuing, I enable Denoise for the render preview in the Render tab.

In physics, a Blackbody is an ideal object that absorbs all electromagnetic radiation and re-emits it with a spectrum, which we'll simply call color, that depends only on its temperature, regardless of the material it's made of. As the temperature increases, it shifts from the red of a glowing stove to yellow-white and, increasing further, the color tends toward blue. In fact, the stars that have the highest surface temperatures are actually blue-white giants.

Regarding the Intensity and Tint parameters: the first has an intuitive meaning and, as mentioned earlier, 1 is a physically realistic value for Intensity, while a value greater than 1 will make the fire brighter, though not physically accurate.

The Tint field lets you change the color of the flames. Of course, this is not a physically accurate representation. For that, you should rely on the temperature value and leave Tint set to white. However, if you want to quickly create colored flames without setting up nodes and parameters with Emission, then this option may be useful.



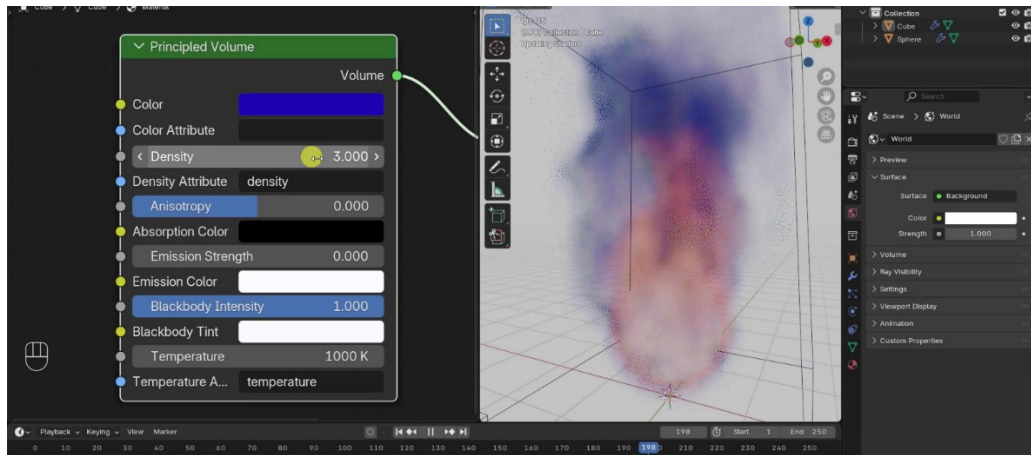
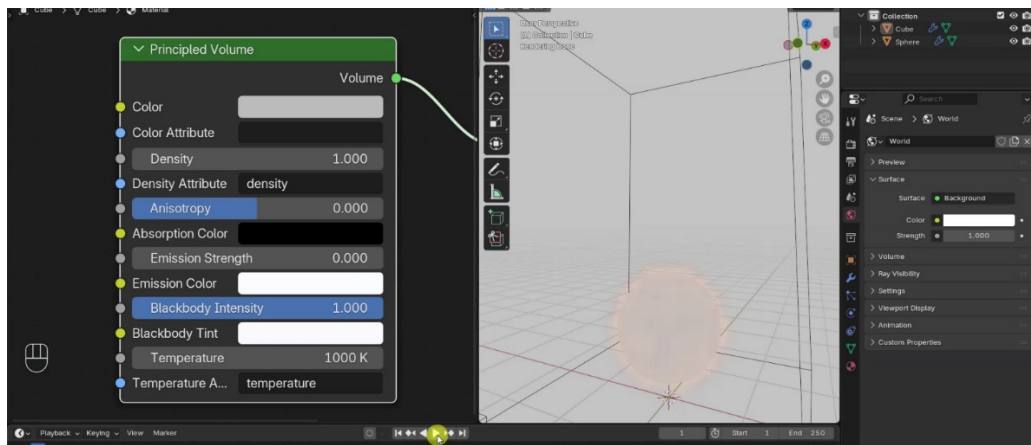
At this point, you're probably wondering how to add Inflow objects with different flame colors, since the color is defined in the Domain Material, not in the Inflow objects themselves.

There are basically two options.

The first is to create separate Domains, but that only works well if the flames don't need to interact with each other. The second solution is to use parameters and attributes from the emitter objects to try to give their flames different colors, but this doesn't always work well. We'll look at a possible solution in a later episode, after covering the other object parameters and Principled Volume in more detail.

At the top of the Principled Volume node, you'll notice a field called Color, with a default gray color. This is the smoke color, since Principled Volume allows us to define both the appearance of smoke and flames within a single node, with distinct parameters. The Density value also refers to the smoke. To show you the smoke, I set the World Background Strength back to 1.

The scattering of light through smoke particles can be adjusted using Anisotropy and Absorption Color. Anisotropy lets you influence the direction of scattering, which can occur more backward or forward depending on whether the value is negative or positive. To make the differences easier to notice, I increased the smoke density by setting the Density value to 2 and framed the object so that the flames are behind the smoke. The Absorption Color field obviously lets you give a color tint to the absorption of light by the smoke.



To talk about the Attributes available in the node, I first need to explain some of the information stored by the Domain, so we'll stop here for this episode.

What we've seen is just the beginning of defining the appearance of fire and smoke during rendering, but between the previous episode and this one, you already have the basic knowledge needed to create and render some flames.

In a few episodes, we'll also analyze the other tools provided by Materials, especially for using Emission, but first we need to cover some other basics of the simulation to better control flames and smoke.

Fire Simulations in Blender 4.5 basics | 3/10: Cache, Baking, Offset

<https://youtu.be/dSrltfFc0rQ>

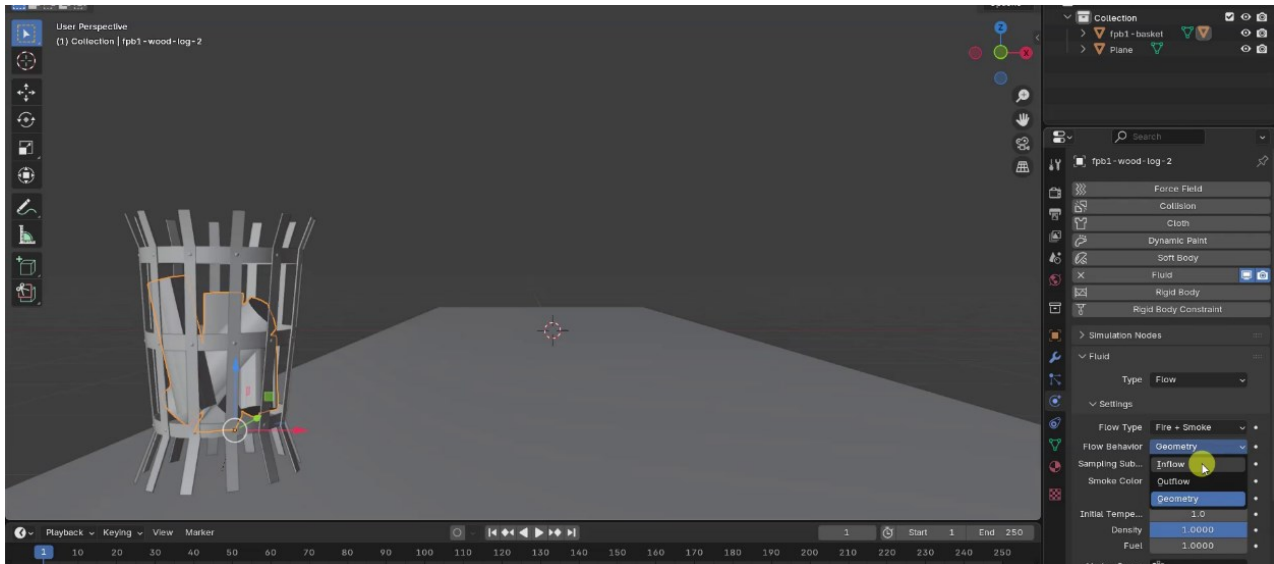
Hello everyone! In this third tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll talk about the ways Blender performs and stores the Bake, that is, the physical simulation of fire and smoke. Among other things, we'll see how to calculate the simulation once for a Domain so it can be duplicated and placed in other parts of the scene without recalculating simulations for all the copies. Finally, we'll see how to set an Offset for the playback of the simulation, so the animation can start with a fire already burning instead of the initial ignition of the flame.

The starting scene for this tutorial consists of a brazier containing some firewood logs.

These are two separate meshes, with the logs forming a single mesh.

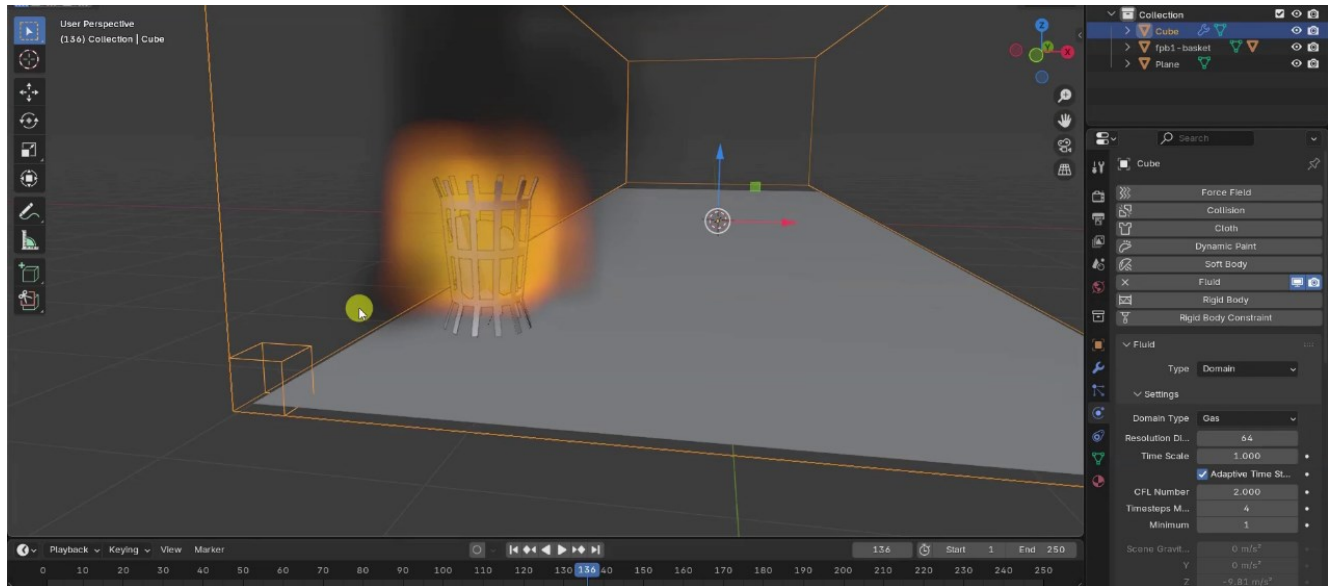
The brazier rests on a rectangular Plane, which serves as the floor, and later we'll place other braziers on it. The virtual world has a Strength of 0 and no light sources, so in Rendered mode the scene appears completely black.

From the previous episodes, we know that in order to make an object produce flames we need to assign it a Fluid component of type Fire And Smoke Flow. So I add a Physics Fluid component to the firewood, setting Inflow as the behavior of the component, which means the wood will generate flames on every frame.



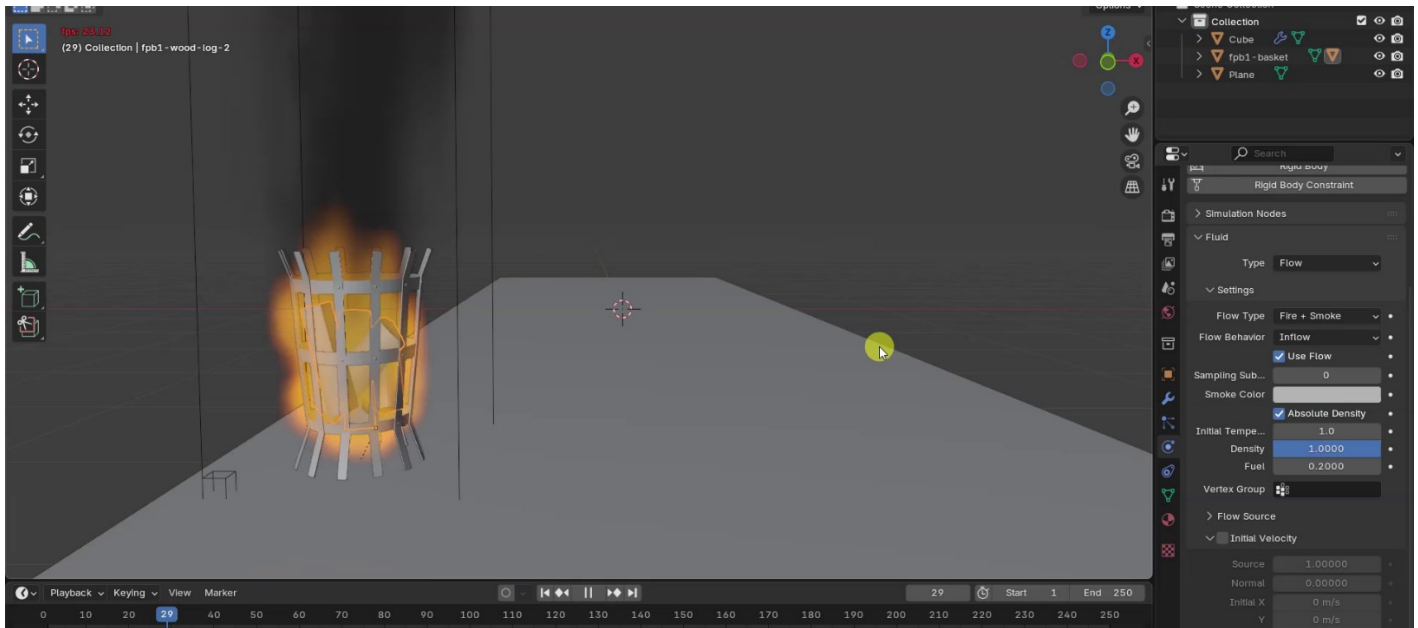
I then create a Cube, assign it a Flow component of type Domain, and have it enclose the entire scene initially. Right away, we notice a problem: the default resolution is too low, producing Voxels that are too large, which makes the flames way too big compared to the objects emitting them.

Increasing the resolution to 64 doesn't improve things much, and the simulation also starts taking longer. I don't even want to try increasing the resolution further, because it's clear this is not the right approach.



The most obvious solution is to resize the Domain so that it only encloses the brazier and a bit of space above it for the flames and smoke to develop. Now the resolution works even a little too well. In fact,

maybe it's even excessive, so we can bring it back to 32 while still getting well-defined flames. The issue, rather, is that the wood is producing far too much fire.



To reduce the amount of fire generated, we can adjust the Fuel parameter of the Inflow object. Intuitively, the higher the Fuel value, the more fuel will be burned each frame. So the solution is to lower this value until the flames reach an appropriate size.

As mentioned in the previous episodes, the data related to Baking, that is the calculation of the physical simulation, is stored in a dedicated folder, whose path is specified in the Cache section of the Domain object. Before creating the Domain, I saved the file to disk, so the cache folder is located in the same directory as the Blender project file.

In the Cache section of the Domain, we can set the frame range over which to calculate the Domain simulation. This doesn't necessarily have to match the animation frame range of the scene. This can be useful for Baking only a limited number of frames just to get an idea of how the fire looks before running the full Bake. Another use case is when Fluid objects are not in the camera view, meaning that for those frames it's unnecessary to calculate the physical simulation.

Before examining the other parameters in the Cache section, let's see what happens when we make copies of the set consisting of the Domain, the brazier, and the logs, with the logs acting as Inflow objects, and place those copies around the scene. As we saw earlier, a single Domain covering the entire scene would need an extremely high resolution to produce good flames, while also leaving a lot of unused space. So, to add multiple braziers, the only option is to duplicate the entire set.

The problem is that when pressing Play in the Timeline, Blender will use the same cache folder for all Domains, which means they'll all share the same simulation data. If you're sure you want to keep the same parameters for all simulations, then perhaps you can try to trick the viewer by slightly rotating the Domains or giving each Domain different Materials, to vary the look of the flames during rendering.

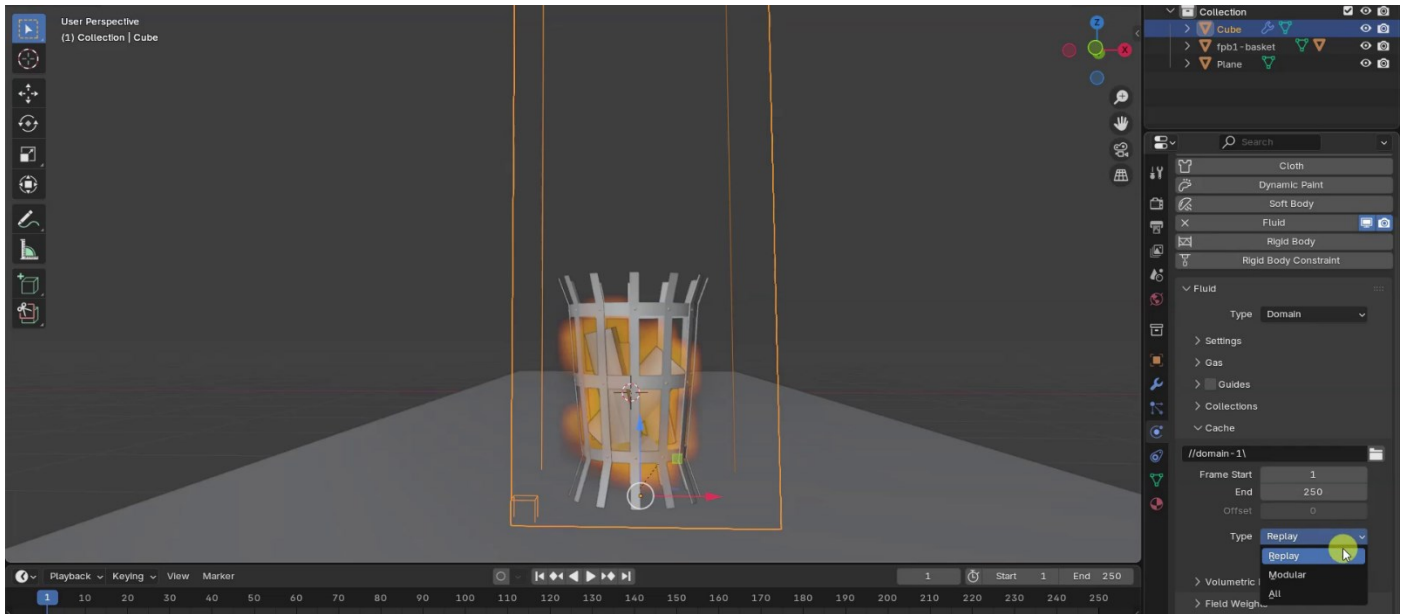
The real problem arises if we change the resolution or any other parameter of a Domain and then run the Bake again. For example, I'm lowering the resolution of the more distant Domains, since we don't need high resolution for background objects. Now Blender is really mixing up the data, because it's the same set of volumetric data being shared by objects that have different characteristics.



If you want each Domain to have its data clearly separated from the others, you need to specify separate cache folders for each Domain before running the simulation. Needless to say, this will significantly increase the disk space used by the data of the various Domains. It's up to you, depending on your needs, to decide which strategy to adopt: one or more very large Domains, duplicating the same simulation, or creating separate caches for the different Domains.

Up until now, Blender has been calculating the Bake of the simulation every time we pressed Play on the Timeline. This is not the only possible mode. In fact, there's another one that also allows us to use a very useful tool. To keep things simple, I'm leaving just one brazier with its Domain in the scene and deleting the others. Be careful, though, because deleting Domains does not delete their data from disk. This is actually a good thing, because it means we can save and reuse the simulation data later. But in this case, I don't need that data anymore, so I'll go ahead and delete it manually.

In the Cache section of the Domain, we find the Type menu, which by default is set to Replay. This is the mode we've been using so far: Blender recalculates a new Bake every time we go back to frame 1 and press Play, overwriting the previous data.

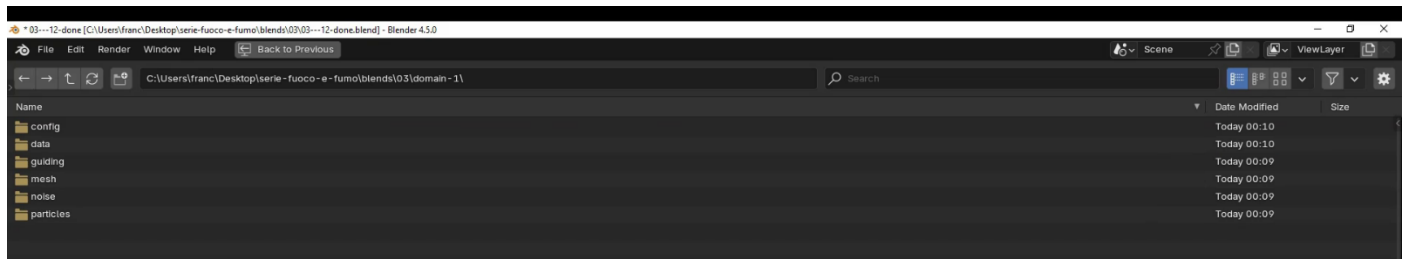


The Modular mode allows you to perform separate Bakes for different aspects of the simulations. This can be useful in some cases, to examine or replace certain parts individually without having to redo the entire simulation.

The All mode, finally, allows you to Bake the entire simulation but calculate it in the background, not in the Timeline. To start the Bake, you need to click the Bake All button in the panel. Blender will display a progress bar for the process. Once the calculation is complete, you can press Play on the Timeline to view the result much more smoothly, since it has already been pre-calculated by Blender.

To modify the simulation settings, you first need to clear the existing data by clicking the Free All button. At that point, the parameters will become editable again and you can change them. Of course, you'll then need to click Bake All to have Blender recalculate the simulation.

Taking a look at the folder being used will give you an idea of the different modules or baking steps that are calculated and stored separately when using the Modular and All modes.



The All mode makes the Offset parameter available, which helps solve a problem you may have already noticed. So far, all the fire simulations started with the flame igniting at the first frame of the animation.

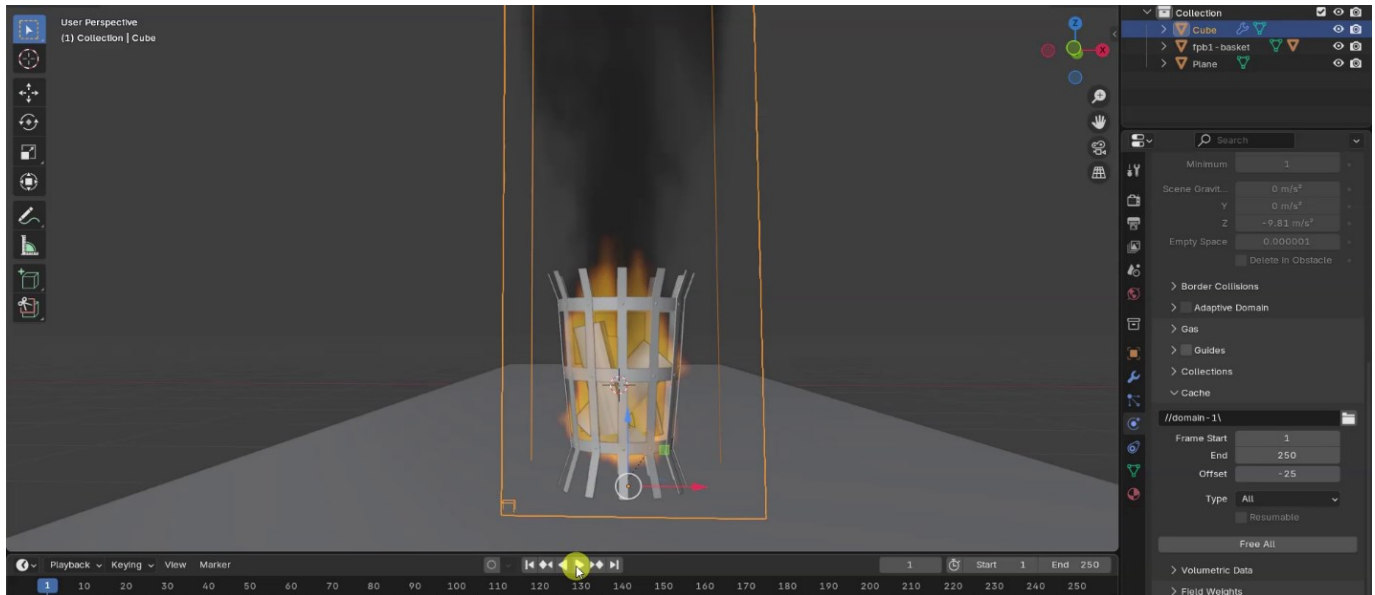
This, however, is not always desirable, because sometimes we may want the animation to start with a fire that is already burning.

To solve this issue, we can enter a negative number in the Offset field.

The tooltip explains that this doesn't mean the Baking will start earlier, but simply that the simulation will be played back as if it had already advanced by that number of frames at the first frame of the animation.

The animation length in frames will still be the one specified between Frame Start and Frame End.

In this case, the last 25 frames of the Timeline remain without simulation, because with the Offset we told Blender to play the 25th frame of the simulation at the first frame of the Timeline.



To solve this, all we need to do is have the Bake calculate more frames so the simulation covers the entire Timeline. So I set Frame End to 275 in the Cache section of the Domain, click Free All, and then run the Bake again by clicking Bake All.

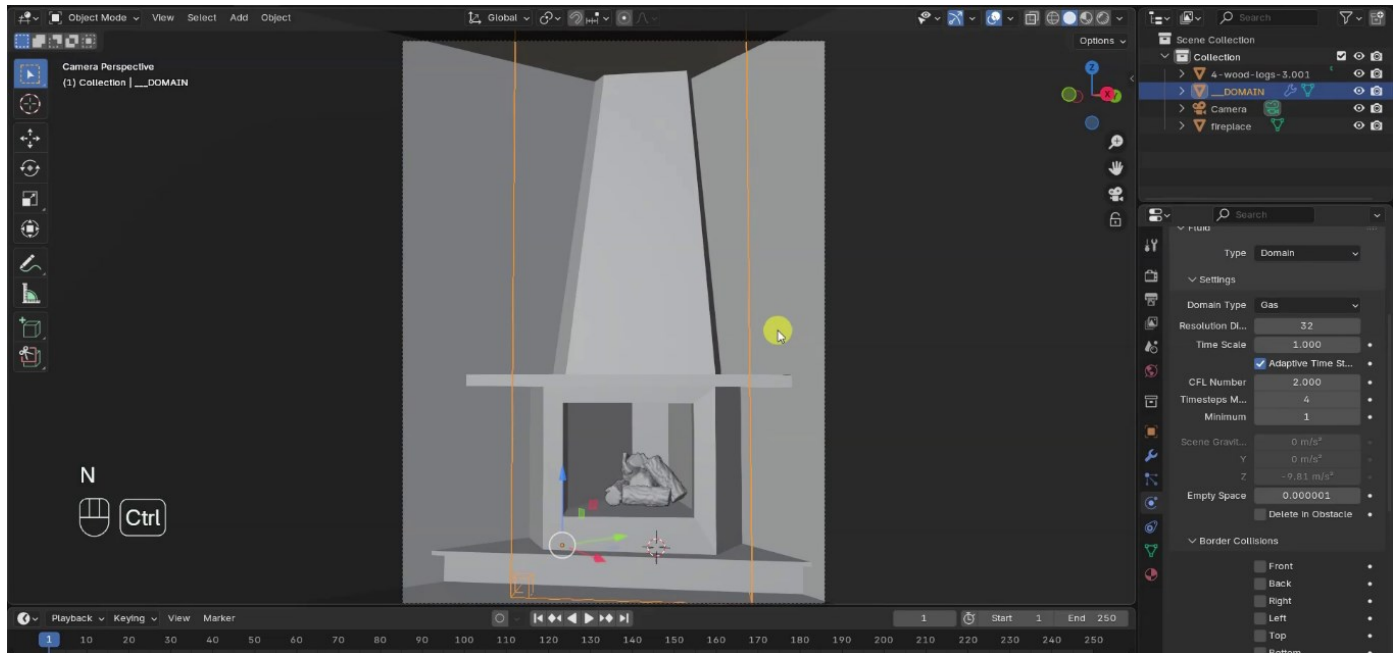
Fire Simulations in Blender 4.5 basics | 4/10: Geometry, Inflow, Outflow

https://youtu.be/yAs_JNIOvl8

Hello everyone! In this fourth tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll take a detailed look at the parameters of the three types of Flow objects used in the physical simulation of fire and smoke.

In the scene I'm using, there are some logs placed inside a fireplace. These logs form a single mesh object, which we'll set up as an Inflow. I've already created a Domain for the simulation, enclosing only the volume of the fireplace.

I assign a Fluid component of type Inflow to the logs. The Fire And Smoke mode includes both the Smoke and Fire parameters, so I choose this combination in order to analyze all of them. Later I'll point out which parameters belong only to Smoke and are therefore not available in Fire-only simulations. As you can see, there aren't too many parameters.



By default, the Flow Behavior is set to Geometry. As we know from previous episodes, this mode produces an initial burst of flame and smoke, after which the object will no longer emit either fire or smoke. It can be useful for explosions or single puffs of smoke, but in our case we need the Inflow mode. In any case, aside from the Use Flow checkbox, Geometry and Inflow share the same parameters, so we'll examine all of them by looking at Inflow.

From the previous episode, you already know that to control the amount of flames produced, we can adjust the Fuel parameter. In our case, we can lower it a bit to generate lively but not excessive flames, or lower it a lot to create glowing embers.



I reduce the size of the Domain a bit and increase the resolution. Before examining the Inflow parameters, let's see how to make the smoke inside the fireplace disappear using an Outflow object.

Outflow objects are meshes that make Fluid simulation elements, whether fire or smoke, disappear upon collision. I'm adding a simple Plane to use as an Outflow, placing it at the top of the fireplace where the smoke would normally rise into the chimney. Outflow objects, like Inflow objects, don't necessarily need to be visible in rendering. So I'm disabling the visibility of the Outflow object both in the 3D Viewport preview and in the final render. From now on, I'll select it in the Outliner.

I then add a Fluid component to the object, set its type to Smoke Flow, and its Behavior to Outflow.

This type of object has very few parameters, and in fact these are also present in Geometry and Inflow objects, so let me briefly describe them.

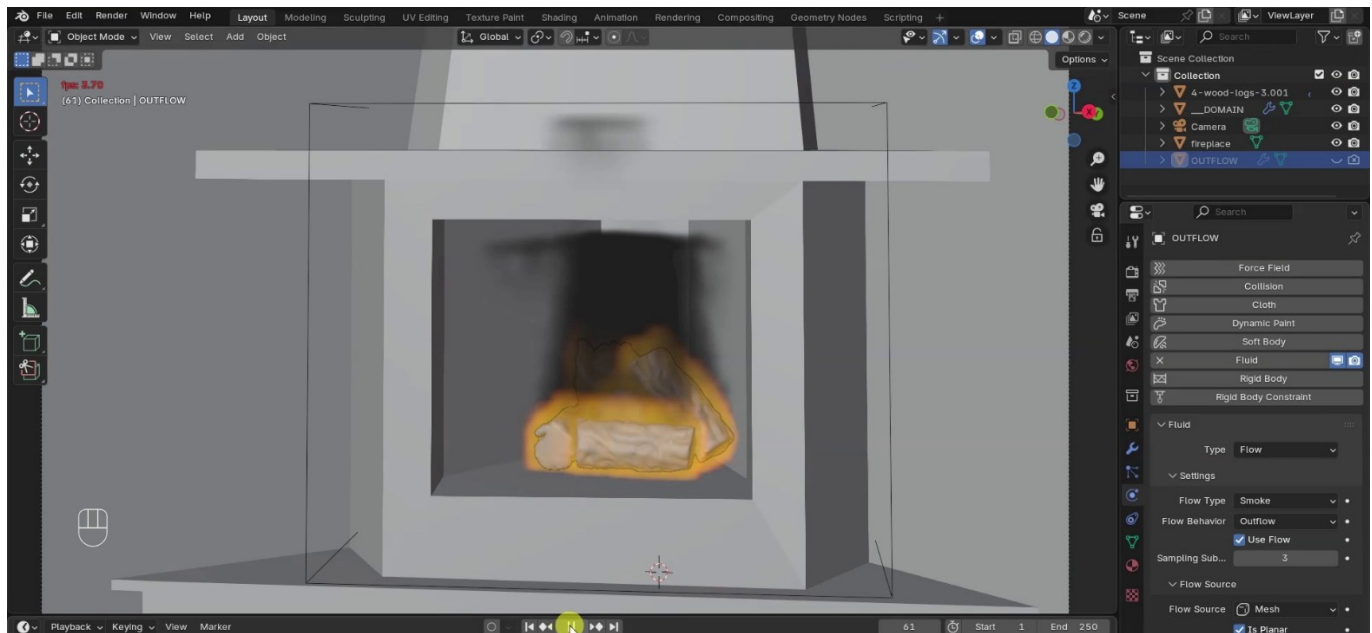
The Is Planar option should be selected if the object has no volume or, more generally, if it isn't closed. Since the Plane is exactly this kind of object, I enable this option.

The Use Flow parameter allows you to enable or disable the Inflow or Outflow, and it can be animated, making it useful if you want to create intermittent or non-continuous emission or removal of fire and smoke in an animation.

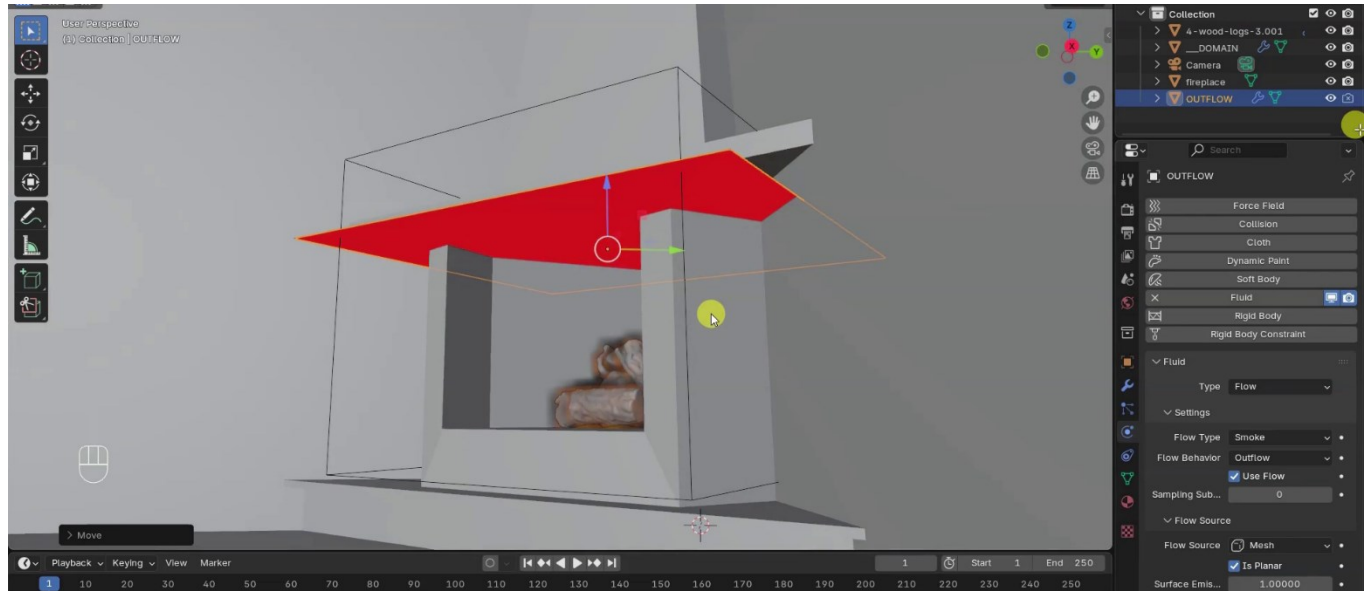
The Sampling Substeps parameter defines the number of additional simulation steps calculated between frames, improving the quality of the simulation. It's very useful for fast simulations, and even a low value, such as 3, produces noticeable differences. Surface Emission and Volume Emission

provide offsets for the emission or capture point of fire and smoke relative to the object's surface or volume, but in this situation we don't need them. We'll talk about Flow Source later.

Playing the simulation, we notice that at first everything looks fine, but at some point a bit of smoke manages to escape upward. Increasing the number of Substeps doesn't change the situation.



Making the Plane visible again in the 3D View and changing the viewpoint, the solution becomes clear: the Plane is too small, so at a certain point some smoke escapes outside and rises. The only solution is to resize the Plane properly and try again.



OK, now let's examine the parameters of the Inflow object in detail, except for those we already covered with Outflow! Smoke Color actually refers to the color of the smoke in the 3D Viewport preview, because the rendering color of the smoke is defined elsewhere. Assigning a different color here can be useful to

identify smoke more easily in complex scenes, or to use this object attribute in Shading to change colors or other aspects of the Domain Material.

The Absolute Density field is enabled by default and generally it's better to leave it that way, because it ensures there is only a certain maximum density of fire and smoke within the Domain.

If disabled, fire and smoke can increase dramatically, sometimes resulting in a kind of uncontrolled blaze. Here, the effect is limited by the presence of the Outflow, so I'll temporarily remove this object from the Domain to show you how the flames increase. Disabling Absolute Density can be useful in some scenarios, but in most cases it isn't, so I'm turning it back on.

The Initial Temperature and Density parameters refer to smoke, so they are not available in Fire-only simulations. Density refers to the amount of smoke to emit, and higher values correspond to larger volumes. In fact, for the scene I'm using, too much smoke is being emitted, so I'm lowering this value.

The Initial Temperature parameter defines the difference between the temperature of the smoke and that of the surrounding environment, and it controls how fast the gases rise. Its effect, however, is tied to a Domain parameter called Buoyancy Heat, so we'll examine it more thoroughly in another episode.

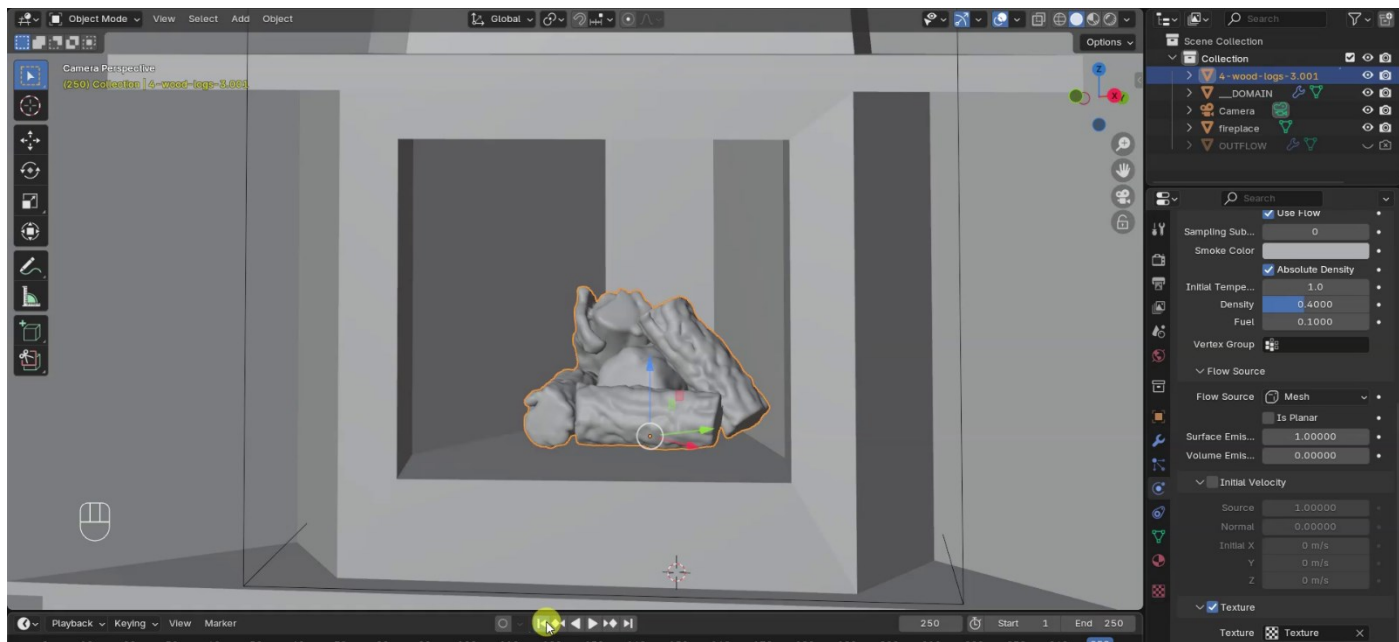
So far, I've made the Inflow objects emit flames across their entire surface, but it's possible to limit the emission to only certain parts of the objects by selecting vertices in a Vertex Group and then specifying the name of this Vertex Group in the Flow component's dedicated field.

Another tool that lets us add variability to flame emission is Texture, found at the bottom of the Inflow panel. Here we can assign a Texture and, most importantly, an Offset field, which defines which "slice" of a procedural Texture to use.

Procedural Textures such as Clouds or Noise are not two-dimensional images but actual three-dimensional volumes.

By animating the Offset value along the Timeline, as I'm showing on screen, we shift the Texture vertically, using a different pattern each time for the emission.

This gives the flames a less uniform look. The issue with this system is that it usually requires multiple attempts to find the right combination of Texture type and Offset change before achieving a good result.



The tools in the Initial Velocity group allow you to give flames an initial direction and speed. Their fields have fairly intuitive names, so there's not much more to say, but I'll still show a practical example with another scene in a later episode.

Finally, we have the Flow Source field, which lets us choose between Mesh and Particles.

The meaning is straightforward: so far, we've used Mesh to emit fire and smoke from objects, possibly limiting the emission surfaces with Vertex Groups.

But objects can also have particle systems, and sometimes we may want those particles, rather than the emitter, to produce fire and smoke.

In that case, we need to select Particles as the Flow Source and, of course, specify which particle system of the mesh will drive the simulation.

As you can guess, we can assign a Flow component either to the geometry or to one particle system of the selected object.

Fire Simulations in Blender 4.5 basics | 5/10: Domain Settings 1/2 (Gas, Fire, Collections)

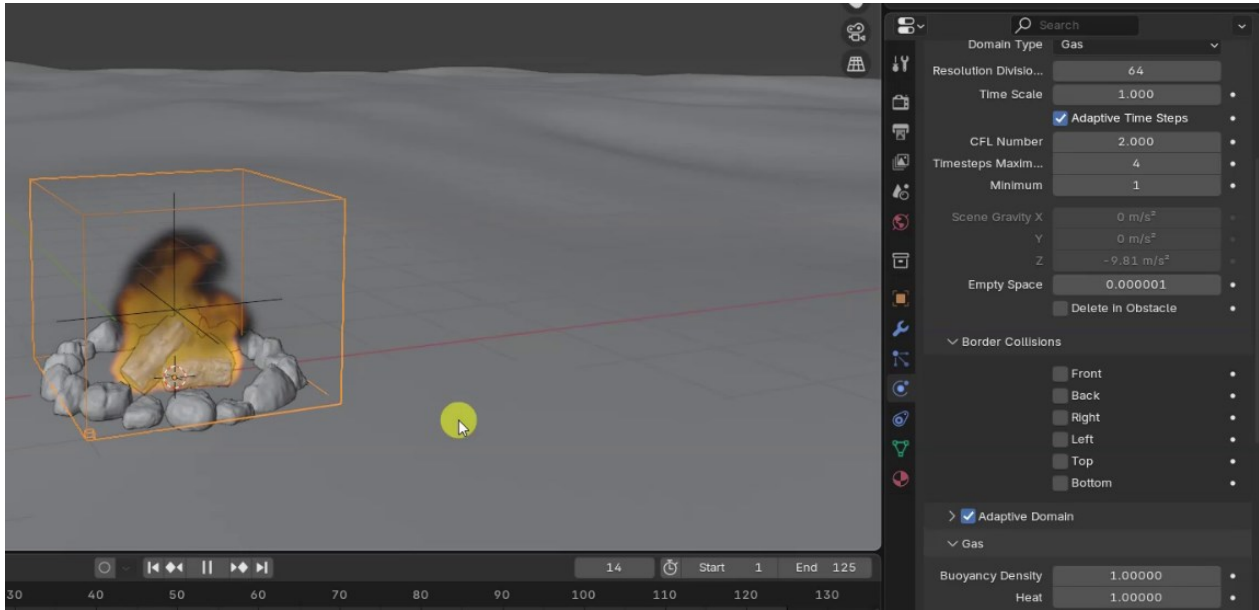
<https://youtu.be/oRiGM-Pl7dk>

Hello everyone! In this fifth tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll take a look at some of the main Domain parameters, specifically those in the Settings, Gas, Fire, and Collections sections. Among other things, some of these parameters will allow us to create flames and smoke that look livelier and less uniform.

Let's start with the main elements of the Settings section.

By now we're familiar with the Resolution Divisions parameter, and we know that if the size of the Voxels shown by the small cube at the bottom of the Domain is very small, while the Domain itself is large, more calculations and memory will be required to run the simulation.

The Adaptive Domain option, found at the bottom of the Settings section, allows Blender to dynamically change the Domain size during the simulation.



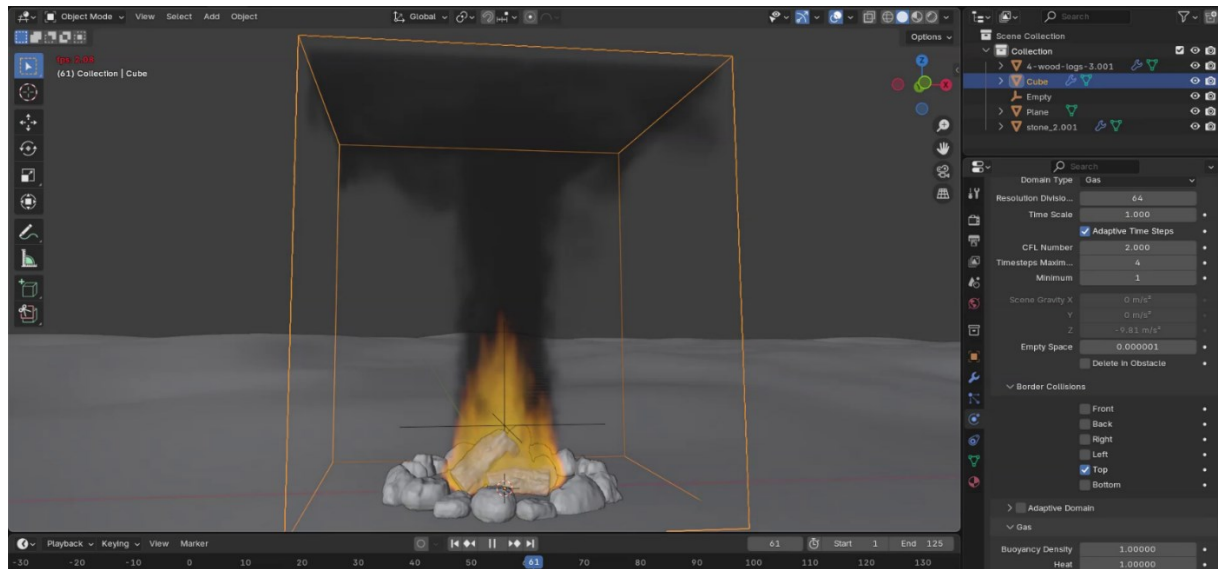
Even with Adaptive enabled, however, the Domain will grow in height only up to a certain point.

After that, the smoke will begin to disappear when it touches the ceiling of the Domain, treating it as an Outflow. It's possible to make the smoke dissolve more quickly with a dedicated option in the Gas section, which we'll look at in a moment.

But first, let's disable Adaptive Domain and take a look at the Border Collisions section.

The Border Collisions parameters, when enabled, cause the sides of the bounding box to be treated as obstacles, deflecting flames and smoke, rather than as Outflows, which is the default when they're disabled, meaning fire and smoke simply vanish instantly at the borders. By enabling Border Collision for the ceiling of the Domain, the smoke will build up there, and as it spreads sideways, it will disappear when it reaches the edges, since Border Collision is disabled on the sides.

Enabling Border Collision on the other sides as well makes the Domain behave like a closed box, where fire and smoke continue to accumulate rather than dissipating on contact with the walls.



Going back to the top of the Settings panel, we find Time Scale and Adaptive Time Steps. Time Scale simply lets you slow down or speed up the simulation.

The Adaptive Time Steps checkbox, which is enabled by default, lets the algorithm decide when to calculate more intermediate steps, called Steps, for each frame of the animation.

The three parameters located under this checkbox relate directly to the algorithm used to determine how many Steps to perform each time.

Maximum and Minimum are self-explanatory, while CFL Number is a bit more complex.

Without going into implementation details, what we need to know is that a higher CFL Number requires fewer Steps and therefore less computation time, but the result will be less accurate.

In simulations where flames and smoke move particularly fast, it's better to use a lower CFL Number to improve the quality of the simulation.

The Gravity parameters are disabled by default because the global settings from the Scene tab are used. If you disable Gravity globally in the Scene tab, these parameters become available again in Physics and can be modified.

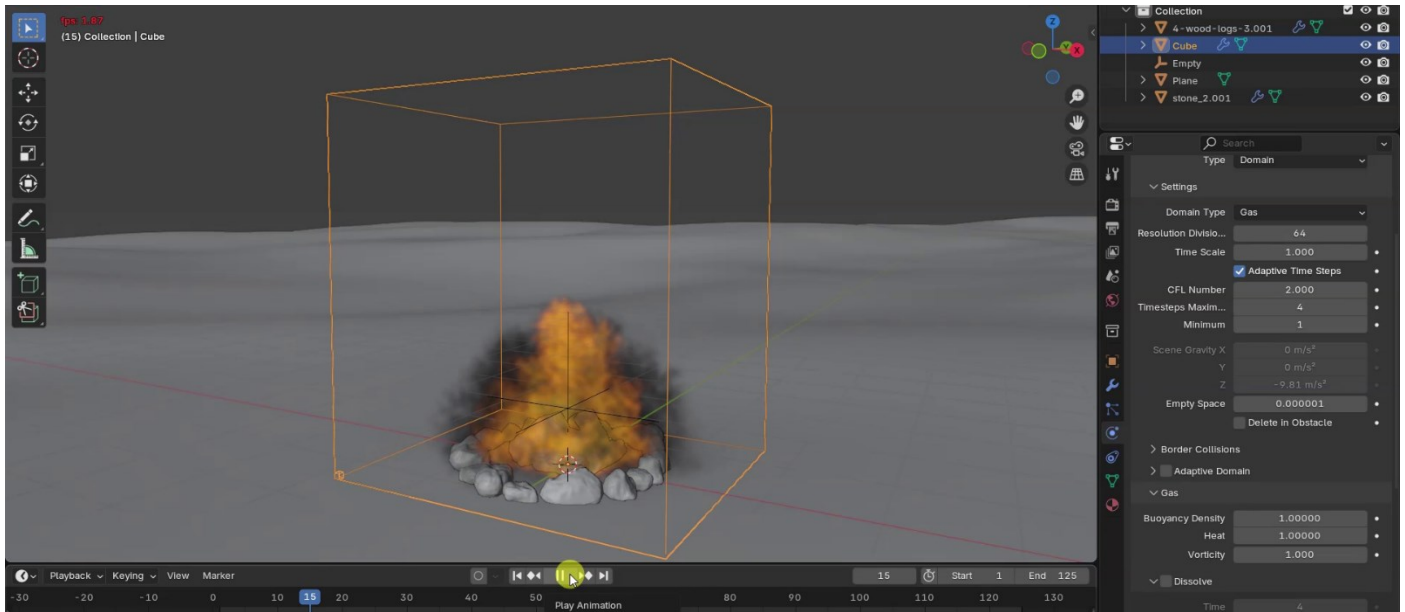
Empty Space works only with Smoke elements. Specifically, Voxels filled below this threshold will be considered empty and ignored in the simulation, saving computation time.

Moving on to the Gas section, we first find the Buoyancy Density and Heat parameters, which represent the upward force on gases, the first based on gas density and the second on their temperature. These two values don't directly represent temperature and density, but rather the upward forces due to them.

Fortunately, the tooltips for these parameters are very clear and explain that increasing these values makes gases rise faster. Buoyancy Heat, in particular, is the upward force due to the heat of the smoke. The initial difference between the smoke temperature and the surrounding environment is defined for each Inflow object using the Initial Temperature parameter. In the Inflow object panel we also find the Density parameter, which is linked to the Domain's Buoyancy Density.

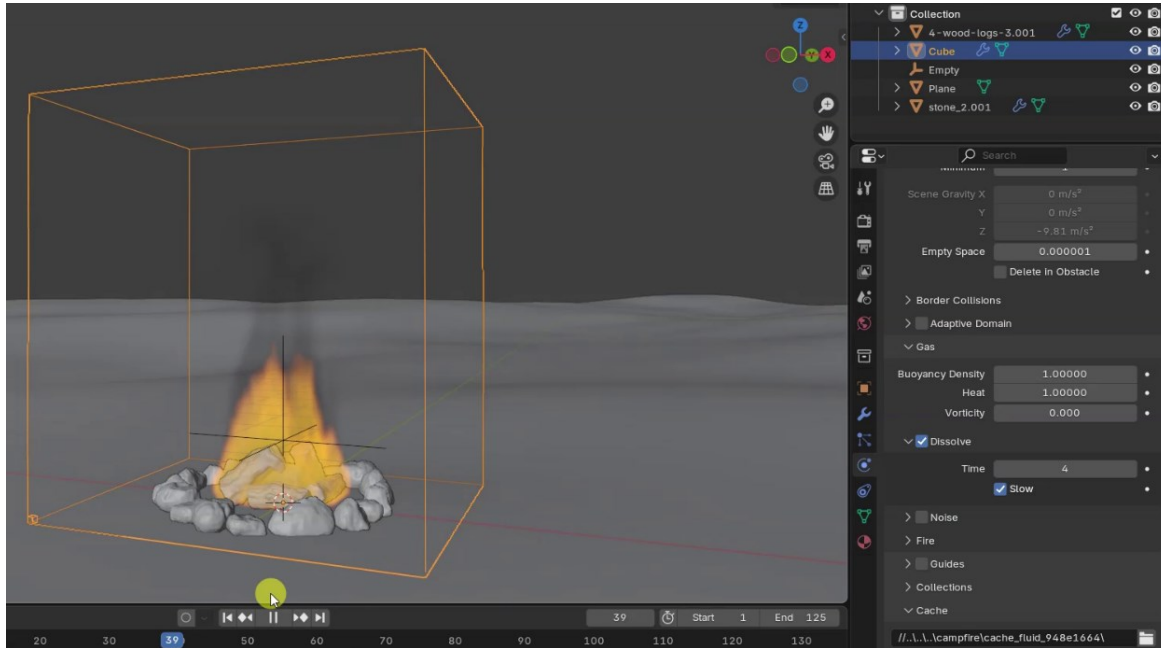
Vorticity controls the turbulence of the smoke, which will also indirectly affect the flames.

The higher the Vorticity value, the more disturbed the smoke will be. Be careful, because even low values for this parameter can create very intense turbulence! On screen I'm also showing how the results vary with changes in Buoyancy Density, which makes the smoke rise faster.



To prevent gases from accumulating on the ceiling, and in general to make them disappear more quickly, we can enable Dissolve and adjust the Dissolve Time parameter, which is measured in frames.

Naturally, the lower the Dissolve Time value, the faster the smoke will disappear, because it will remain visible for fewer frames. The Dissolve Slow option makes the gases dissolve logarithmically, that is, quickly at first and then more slowly over time.

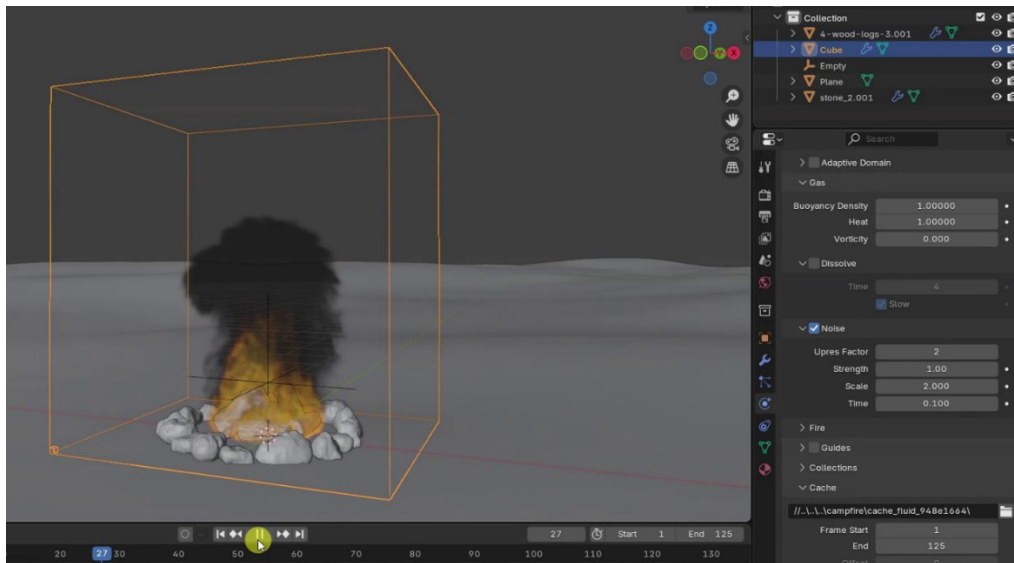


The Gas section also includes Noise and Fire, which allow us to add disturbance to the flames. Noise, in particular, mostly affects the flame contours, adding lots of detail there, making it useful for creating, for example, a campfire or fireplace fire that looks more turbulent and less uniform.

The Strength and Scale parameters intuitively control the intensity of the disturbance.

The Upres Factor parameter refers to an additional subdivision factor applied to the Domain's base resolution, used to create noise details. This also explains why the simulation slows down, since the noisy regions are calculated at higher resolution.

As for the parameters in the Fire section, we immediately notice that there is also a Vorticity parameter here, just like in the Gas section. As with smoke, Vorticity introduces turbulence, in this case to the flames. Flames also have a temperature, and as with smoke, higher values make them rise faster. Combining the two parameters results in taller, livelier flames.



The Collections section contains two fields where you can specify Collections, called Flow and Effector. These allow you to include in the simulation only the Flow and Effector objects that belong to the specified Collections, if any.

Effectors are special elements that let you define objects capable of deflecting the path of fire and smoke. We'll cover the two types of Effectors, Colliders and Guides, in a later episode.

By default, no Collections are specified, so all Flow objects and all Effector objects are taken into account in the simulation.

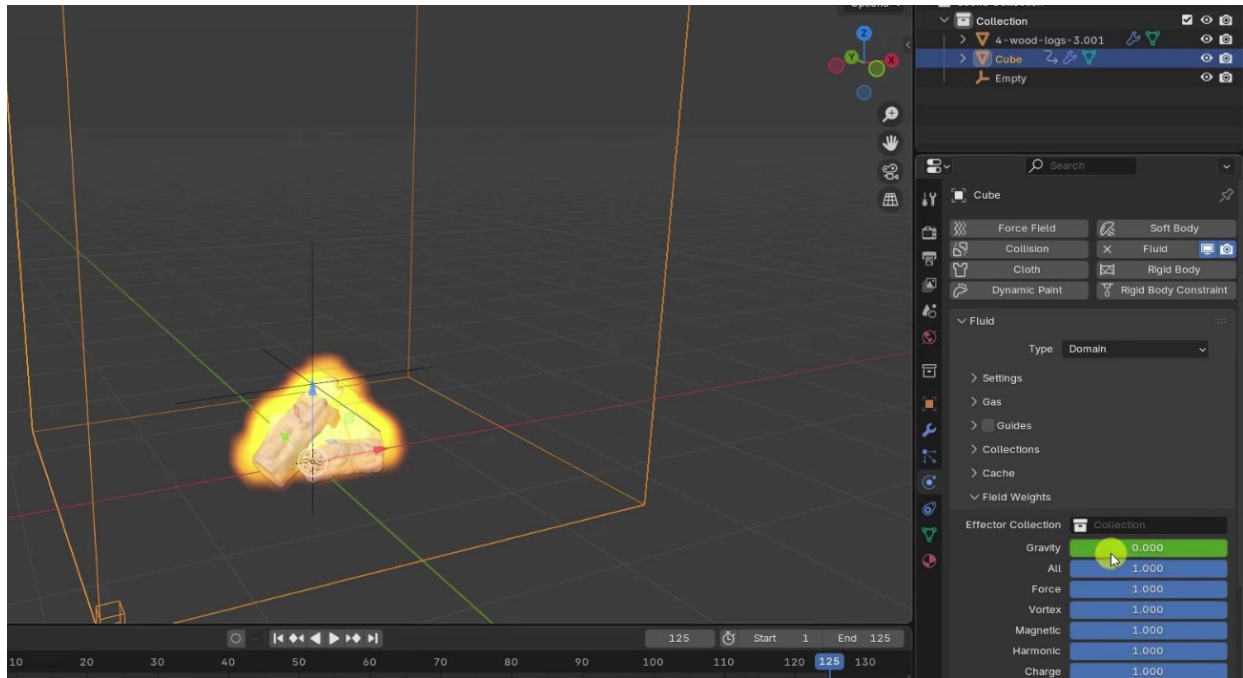
Fire Simulations in Blender 4.5 basics | 6/10: Domain Settings 2/2 (Weights, Data, Render, Display)

<https://youtu.be/obBL5RbReAI>

Hello everyone! In this sixth tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll complete our overview of the Domain object settings. In particular, we'll go through the main parameters of the Field Weights, Volumetric Data, Viewport Display, and Render sections. Some of these parameters will help us better understand certain Material properties, called Attributes, which we'll cover in the next episode.

Regarding Field Weights, for now all we need to say is that the simulation can be influenced by external forces, and here we can define the extent to which these external forces affect the simulation elements.

One of these external forces is gravity, which is enabled by default at the scene level and is what actually makes fire and smoke rise upward. So, you can get a sense of how these fields work by adjusting the Gravity value in the Field Weights section. Among other things, the Field Weights parameters are animatable, which means you can modify the influence of a given force during the animation from this panel, without changing the force's own settings.



There is also a control called All, which lets you apply the same value to all force fields affecting the simulation. The Effector Collection field lets you restrict the simulation to only the Effectors belonging to a specified Collection. By default, it's empty, meaning all Effectors within the Domain will influence the simulation. In any case, we'll talk about force fields and Effectors in later episodes, so for now we'll close this section.

When I talked about the cache earlier, I focused on what information was stored and how to reuse it, but not on how it's stored.

These parameters are found in the Volumetric Data section of the Cache panel, and they let you specify the format used to store Voxel data on disk, along with compression and precision settings.

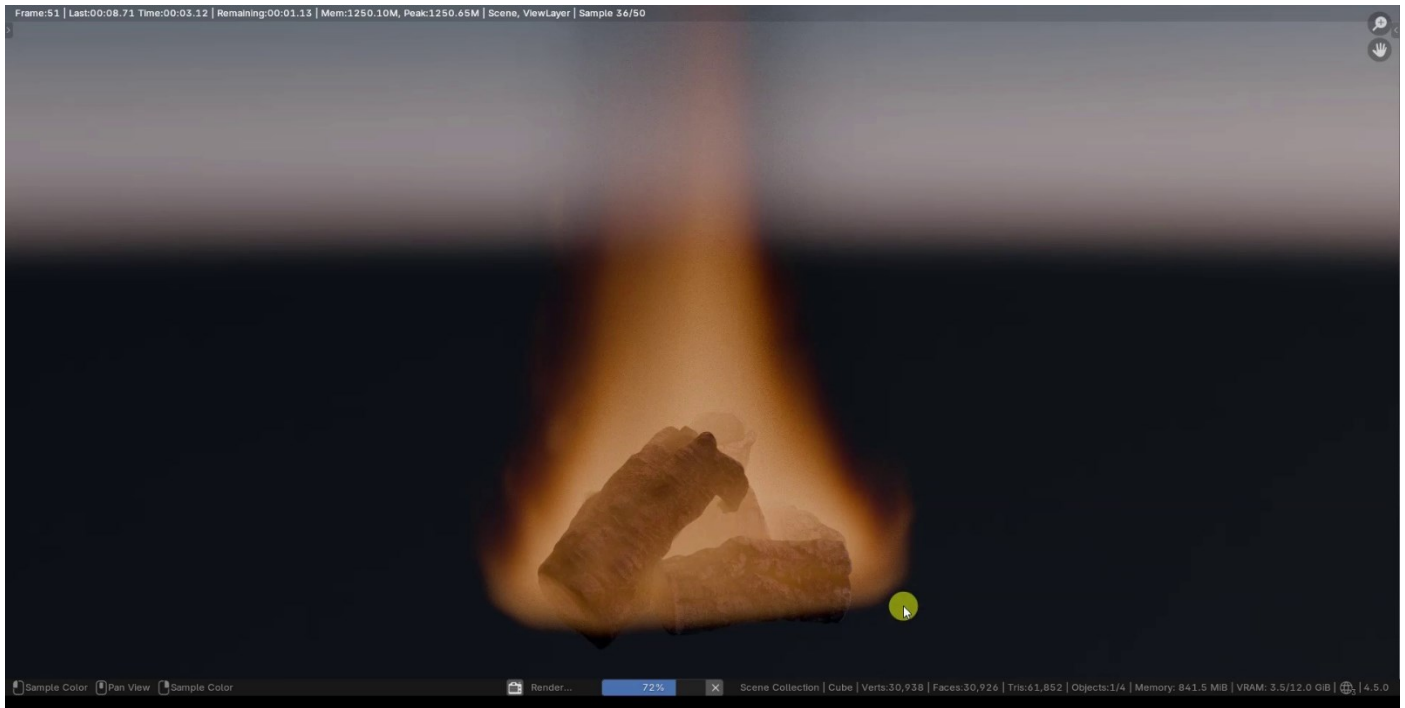
The best compression for disk storage is Zip, but the default option is Blosc, since it's multithreaded and therefore faster.

Half precision saves data at 16 bits, while Full saves at 32 bits, which may be excessive in some cases.

The default settings are Open VDB, Blosc, and Half. In most cases, these defaults work just fine.

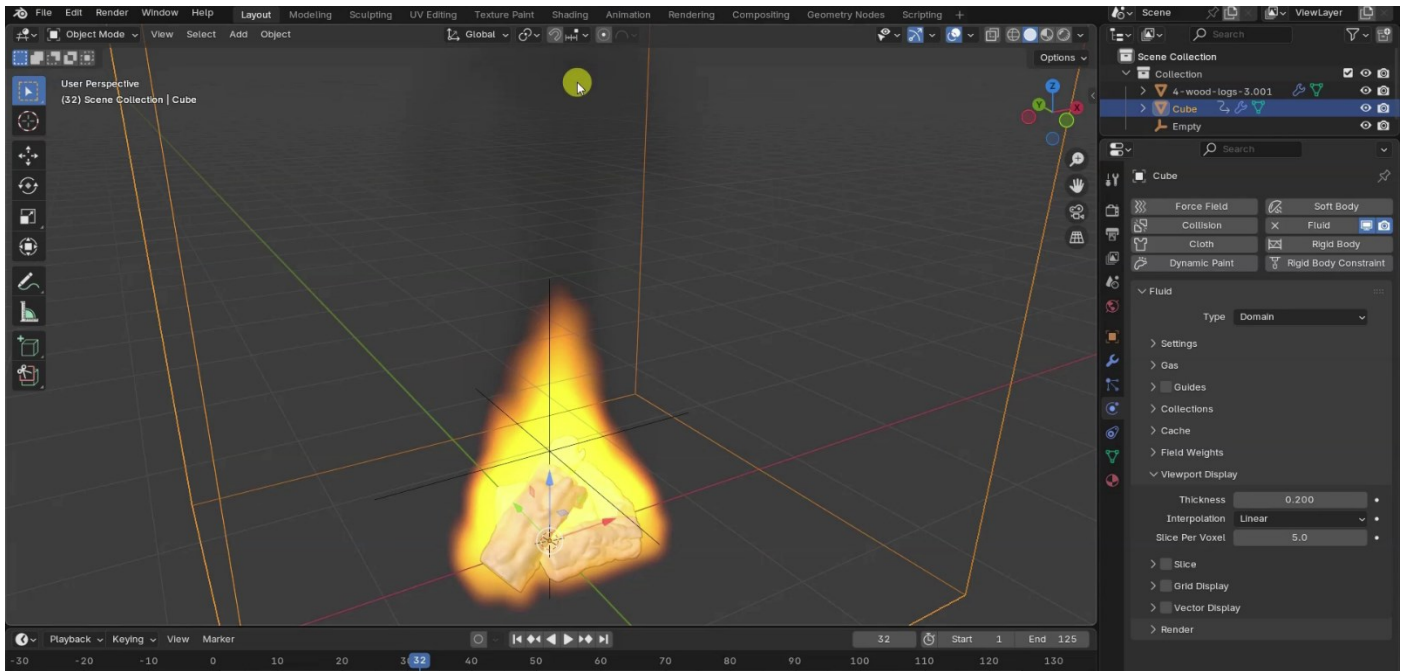
In the Render section, there's actually only one parameter, which relates to Motion Blur.

Motion Blur must be enabled at the project level in the Render tab, and as we'll see in a rendering tutorial later on, in many cases it can produce a very pleasing effect.



The core of this tutorial, however, is the Viewport Display section, which contains many interesting options that let us visualize some of the information contained in individual Voxels. This will help us better understand certain Domain Attributes that we'll use in the next episode to modify fire and smoke materials.

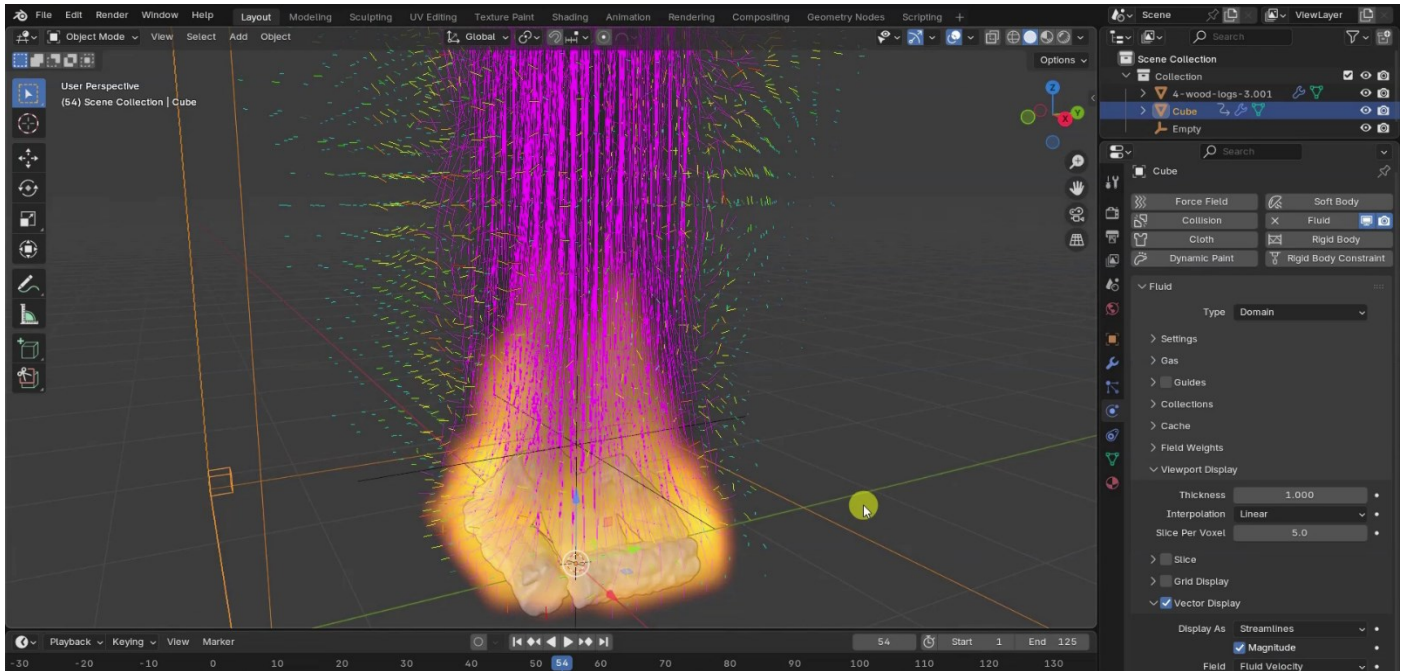
To begin, you should know that the 3D View displays multiple slices for each Voxel. Here you can specify parameters such as smoke thickness, interpolation quality, and the number of slices to calculate per Voxel. Increasing the smoke thickness can make it stand out more, making its behavior easier to observe. Keep in mind, though, that the parameters in this section only affect how the simulation looks in the 3D Viewport in Solid mode, and do not affect its appearance in the final render. They are purely for visualization.



If you want to see a single slice of the entire Domain, you can enable the Slice option and choose an axis. It's like cutting the Domain along that axis and viewing its cross-section. At first, the display will change with every camera movement in the 3D scene, because the slicing axis is set to Auto by default, adapting automatically to the observer's viewpoint. The Position value ranges from 0 to 1 and corresponds proportionally to the size of the Domain along the chosen axis. I'll show you an example using the Y axis and adjusting the Position value.



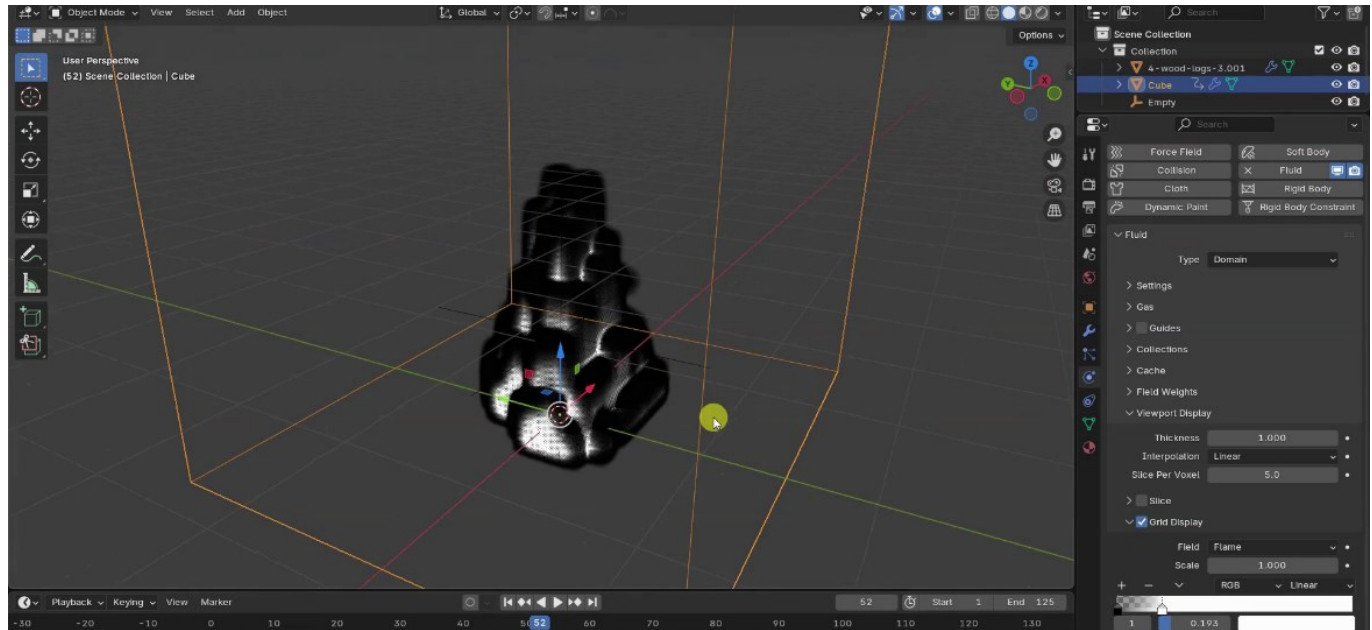
Vector Display shows, within each Voxel, the vectors of Velocity, Guides, and Forces. Guides are special objects that, like Force Fields, influence the behavior of fire and smoke. At the moment we don't have any Guides or Forces in the scene, so we can only see the Velocity representation, which is inherent to the fluid itself. The other options in this section allow us to adjust the look and size of the graphical elements used for visualization, making them easier to read.



I've saved the Grid Display section for last, since I think it's the most interesting.

It lets us visualize some properties of the simulation, such as heat or density for each Voxel.

As we know, the Domain is divided into Voxels, and the resolution is clearly visible from the cube subdivisions we see in Grid Display.



For each Voxel, the Domain contains information such as temperature and density.

By adjusting the Color Ramp, we can see the range of values for these properties across all Voxels.

This kind of visualization is particularly useful because some of these properties are stored as Attributes accessible in the Shading section.

In the next episode, we'll see how to retrieve them there to modify the Material of fire and smoke Voxels, for example based on their temperature!

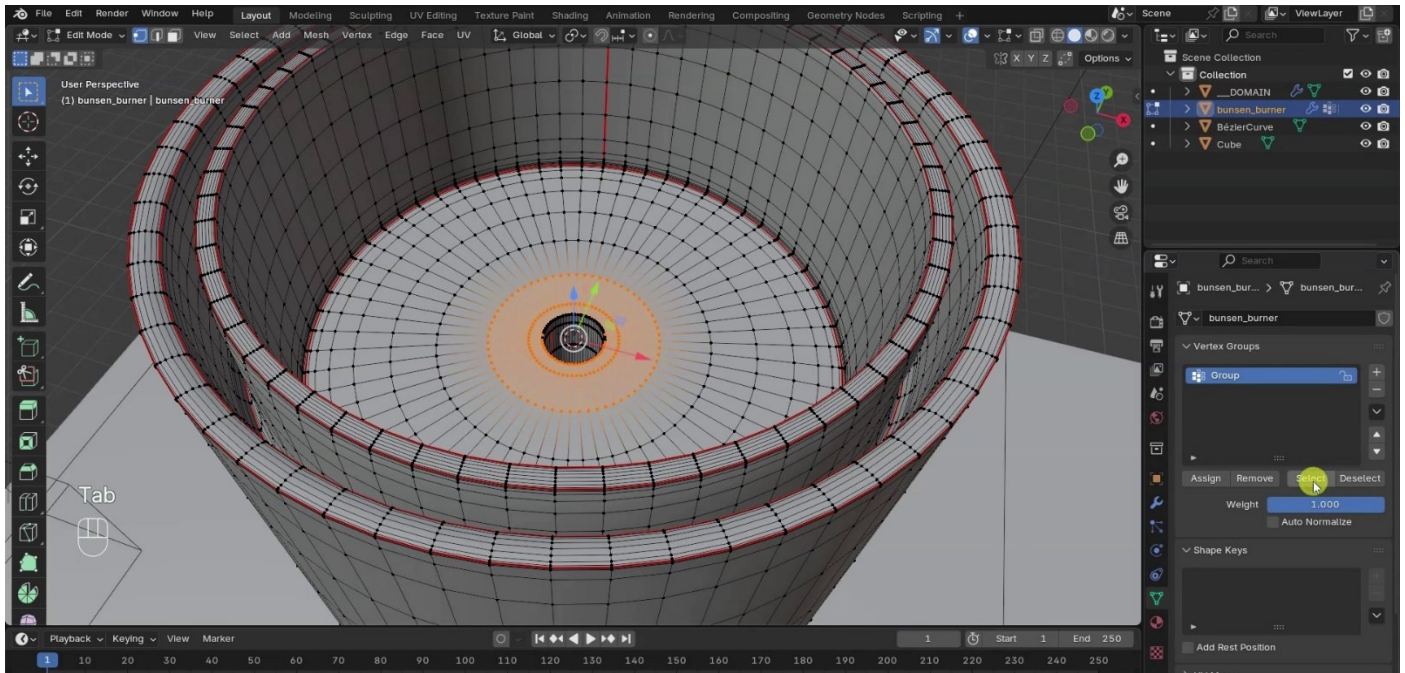
Fire Simulations in Blender 4.5 basics | 7/10: Emission, Attributes, Volume Info

<https://youtu.be/v2dyypR2MX8>

Hello everyone! In this seventh tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll see how to use Voxel Attributes, described in the previous tutorial, to modify fire and smoke materials for rendering.

In the scene I'm using for the first example in this episode, the fire emission comes from a Vertex Group of the object with the Fluid component set to Inflow Fire. The Domain of the simulation encloses only the upper part of the burner, since that's where the flame develops.

Speaking of emission from a Vertex Group, if the emitted flames are not visible, the issue may be related to the Domain resolution. With resolutions that are too low, and therefore Voxels that are too large, the Voxels containing the emitting vertices might be considered empty and fail to emit flames. So try increasing the Domain resolution.



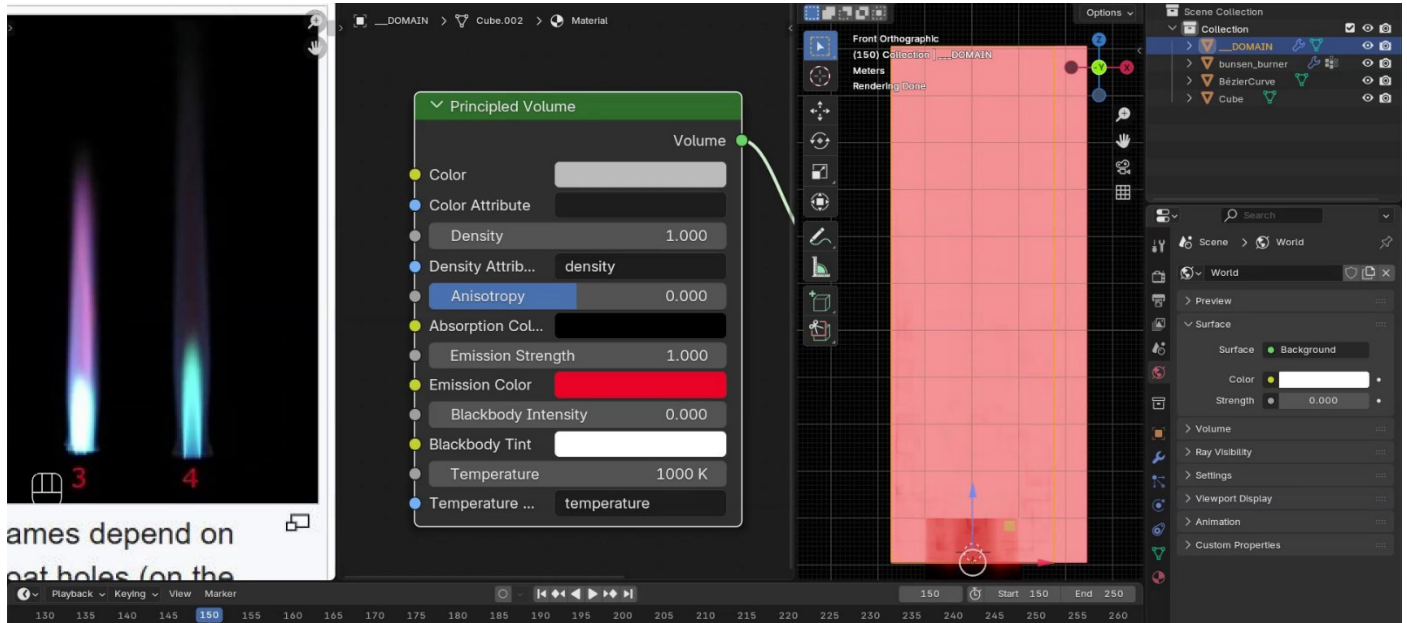
With the default settings, the initial flame will be very low, even invisible from a side view, but by now we know several ways to adjust its appearance. For example, we can act on the initial velocity of the Inflow object to give the flames an upward push, or on the Domain's reaction speed to make the flames taller. Regarding the type of flames produced by this burner, they can vary in both turbulence and color.

To obtain a fourth-type flame, with almost no turbulence, I lower the Vorticity value in the Domain's Fire section to 0.

As a last change before moving on to defining the Material, I'm adding more vertices to the emission Vertex Group and limiting the Timeline animation to the last 100 frames of the simulation, that is, when the flame is already well developed and stable.

OK, let's move on to defining the Material! I'm setting the 3D side view directly to Rendered mode with the World background Strength set to 0, so at first the preview will be black. In an Image editor on the left I've placed a reference image, while in the center I have the Material editor with the Domain, which has a Principled Volume node.

As we know from previous tutorials, increasing Emission Strength and assigning a uniform color to the Emission Color field gives us a very disappointing result. We've seen how to use Blackbody, but now it's really time to learn how to use the Emission mode.

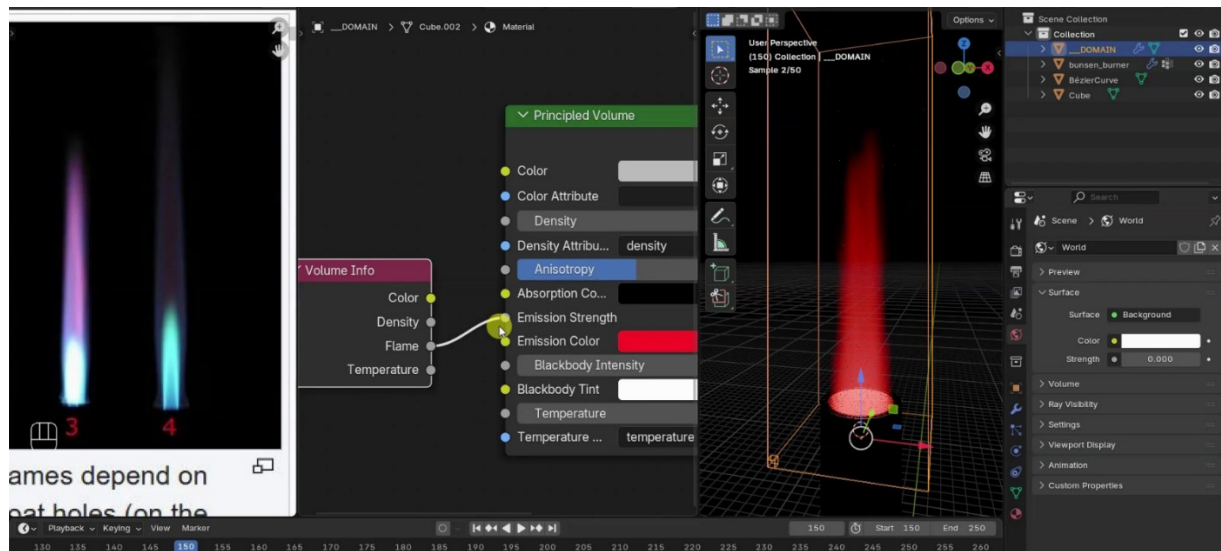


From the previous episode, we know that the fluid simulator stores information for each Voxel. These data can be retrieved using Attribute nodes, by typing the attribute names in the appropriate field. However, volumetric materials have a dedicated node called Volume Info, which provides these attributes as output sockets.

By connecting the Flame output or the Temperature output to the Emission Strength input, the appearance of the flame changes dramatically! The Flame output provides better interpolated values compared to the Temperature output, which also makes the Domain resolution more evident.

For this reason, I'll use Flame.

The Flame and Temperature outputs can also be used for Emission Color, especially if we place a Color Ramp node in between. This way, we can change the flame's color at different points based on the information contained in the Domain Voxels.



I'm using the Flame information for the Color Ramp and therefore for the Material's Emission Color parameter. At this point, the task is to define the color and transparency levels of the Color Ramp's stops. For this, you can use Blender's Color Picker to sample colors directly from the reference image.

Keep in mind that the colors of the various stops can also include transparency levels, which are very useful for the outer parts of the flame, where you often see a semi-transparent region before the boundary, which is more opaque.

You'll also need to carefully choose the interpolation between the Color Stops.

Among these options, the most extreme is Constant, which switches abruptly from one Color Stop to another. It isn't very realistic but can be used for more artistic representations.

If this tutorial on Materials seems short, it's only because we're building on the information and parameters we studied in the previous tutorials.

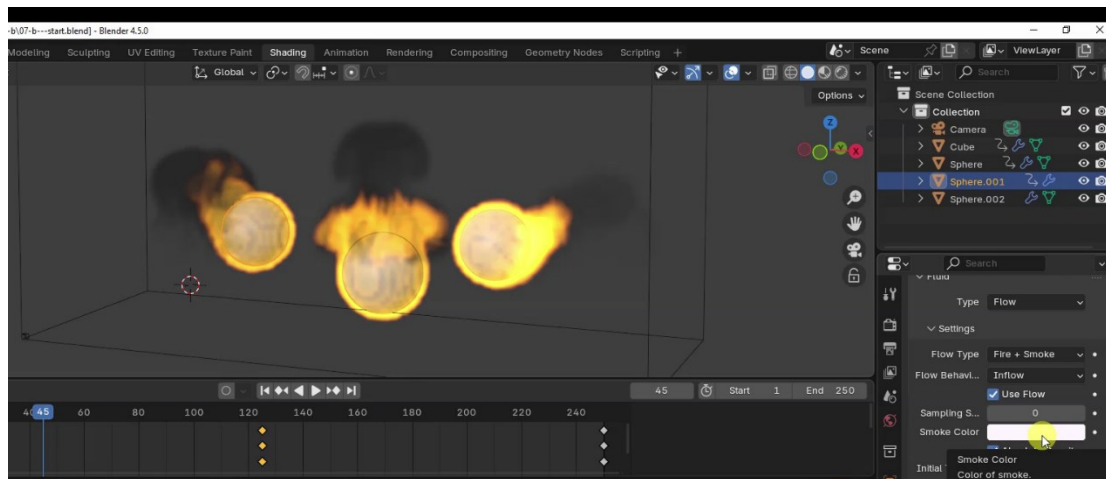
At this point, we already know enough to start defining different flame shapes through the Fluid parameters of both Domain and Flow objects, and their visual appearance using Blackbody or Emission, with the various nodes provided by Blender.

Before wrapping up this tutorial, let me show you how I created three Inflow objects with three different colors inside the same Domain. The result isn't perfect, but I'll use it as an opportunity to demonstrate another use case for nodes. In particular, the Color Ramp.

As we know, the Material must be defined at the Domain level, because it is generated from the Domain's Voxels. This means we can't assign different Blackbody nodes to the Inflow objects inside it.

However, one of the fields of the Inflow objects is Smoke Color, defined in the Inflow panel.

This color can be retrieved in the Domain Shader through the Volume Info node, specifically by using the Color output.



Using this field as the Factor in a Mix Shader node, we can assign different Principled Volume nodes to different Voxels depending on the smoke color defined for them.

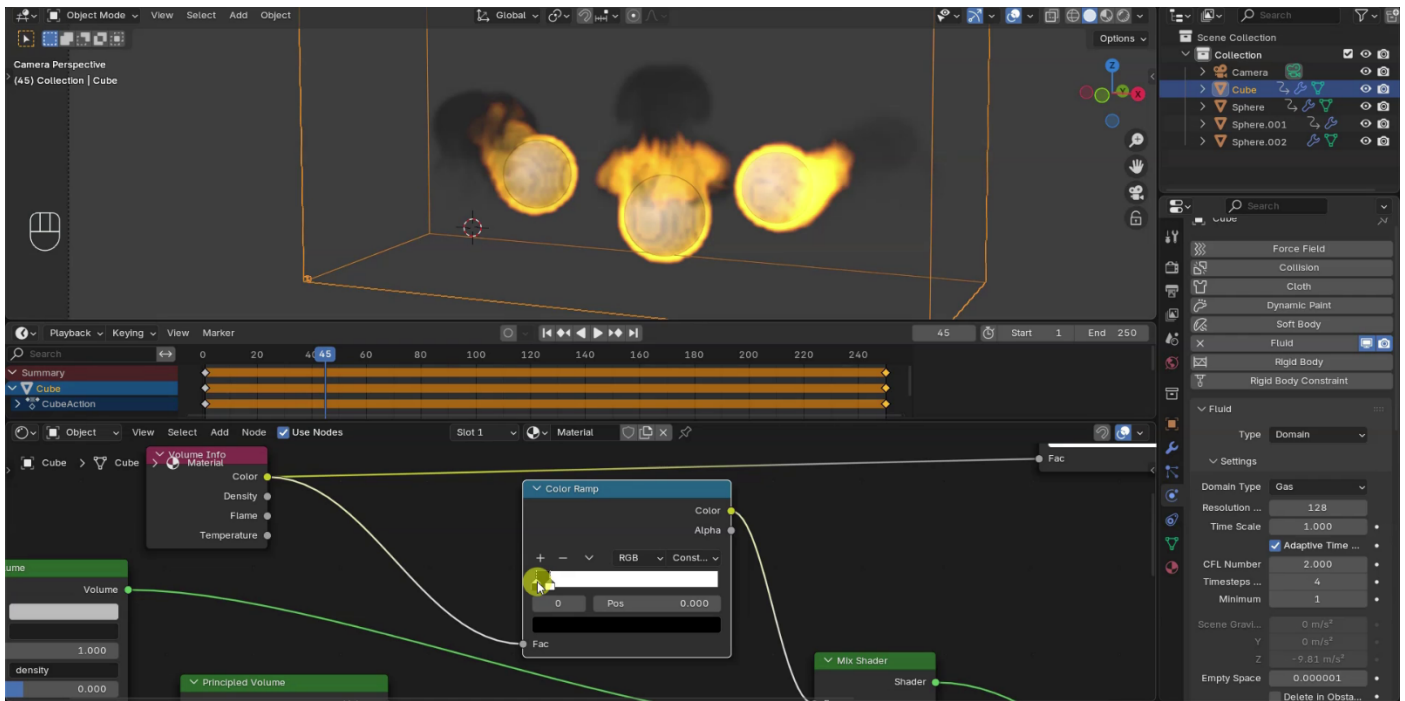
But the Mix Shader alone would simply blend the two Shaders fifty-fifty, so we need to use a Color Ramp with a sharp separation between the colors coming from Volume Info.

To achieve this, I set the interpolation method to Constant.

I then assigned three different colors to the Color fields of the three Inflow objects: white, gray, and black, from left to right.

The first Color Ramp separates black from the other colors. Black corresponds to 0, so it is sent to the first Shader input of the first Mix Shader node, which in this case is connected to a Principled Volume node with a red Blackbody Tint.

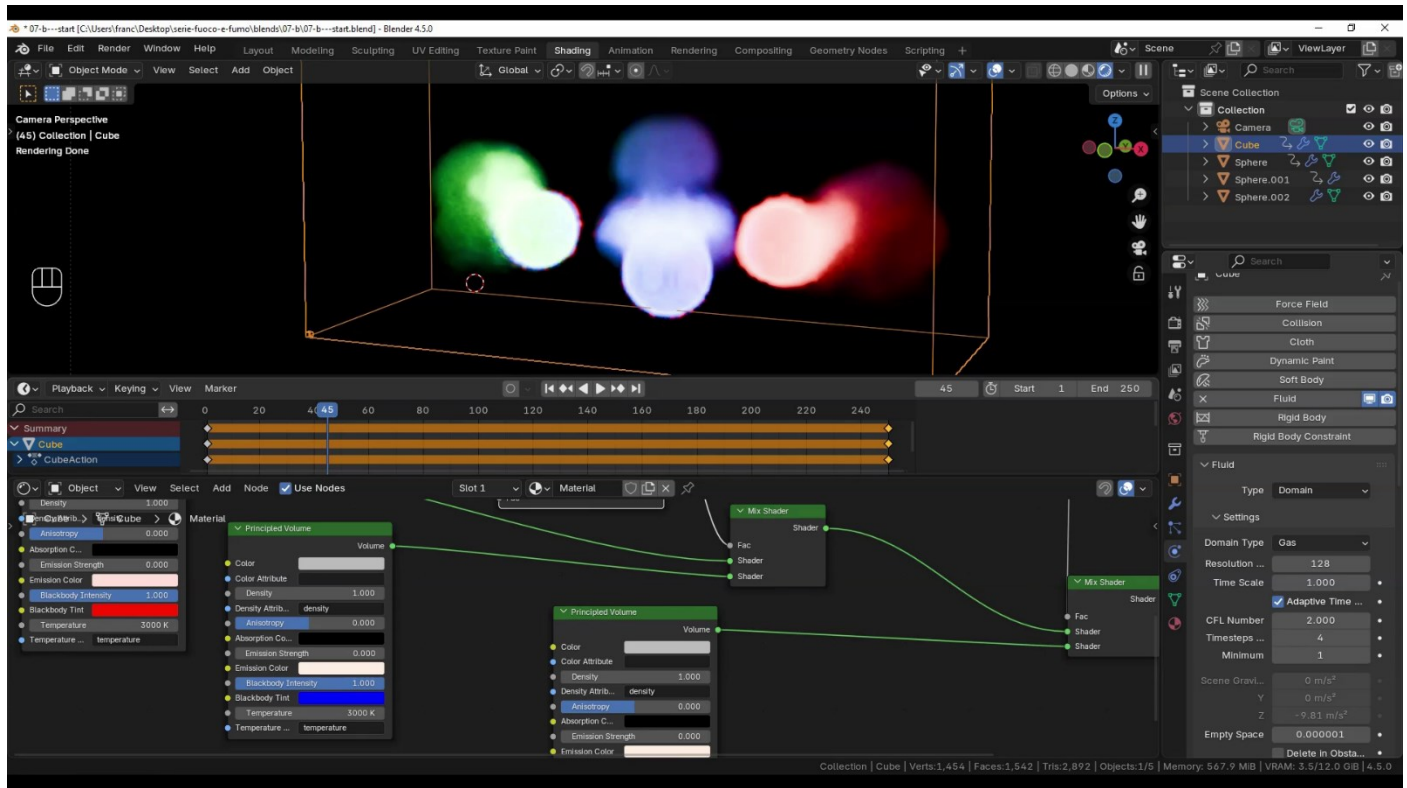
I'm also enabling Denoise in the 3D preview to better define the color of the elements in Rendered view.



The second Shader input of the Mix Shader is connected to a Principled Volume node with a blue Blackbody Tint. So, at this point, if the smoke color is black, we get a red Blackbody.

If the smoke color is not black, we get a blue Blackbody.

Next comes another Mix Shader node, which takes as input another Color Ramp, set specifically to Constant with a Color Stop placed right at the midpoint of the ramp.

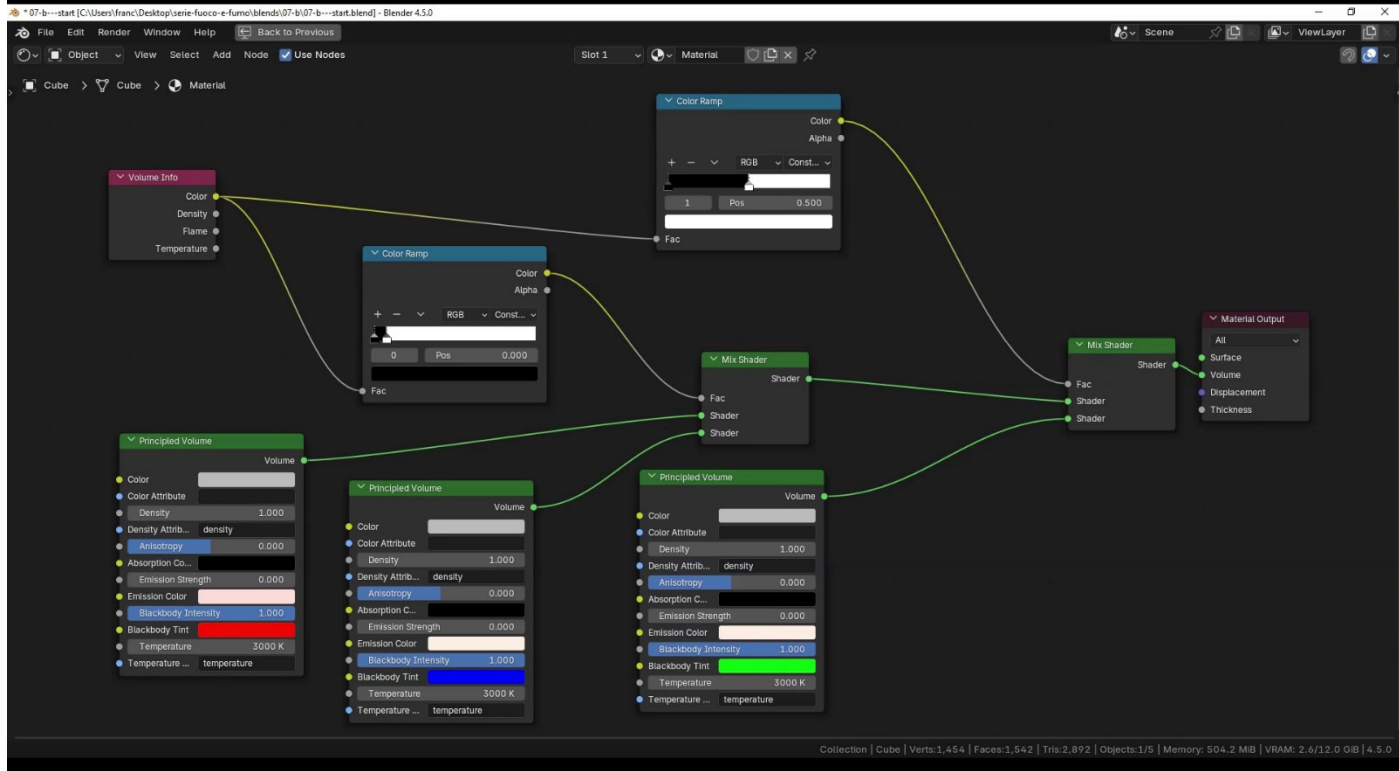


In this case, if the smoke color is between black and gray, the first Shader input of the second Mix Shader is used. That Shader was the result of the earlier distinction between black and gray, so the final color will be either red or blue.

If instead the smoke color is between gray and white, then the second Shader input of the second Mix Shader is used. This input is connected to a Principled Volume node with a green Blackbody Tint.

So, the object with white as its Smoke Color will be rendered in green.

Of course, this isn't the only method you can use to create fiery spheres with two or three different colors, but I wanted to show it as an example use case. As I mentioned earlier, the separation is not perfect, so I encourage you to suggest tweaks or alternative setups in the comments if you find others!



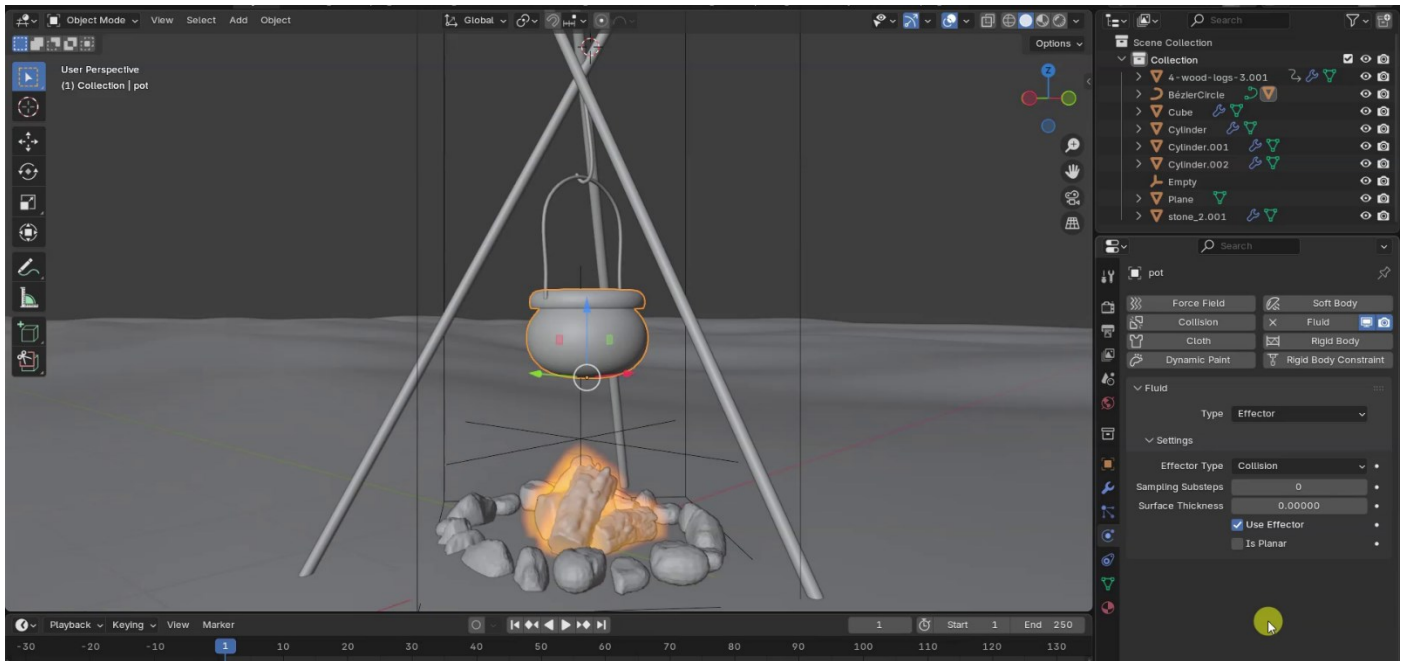
Fire Simulations in Blender 4.5 basics | 8/10: Effectors (Guides, Colliders)

<https://youtu.be/Y4MV2-LmpcE>

Hello everyone! In this eighth tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll see what the two types of Effector objects available in Blender are, namely Collision and Guides.

Effector objects are used to deflect elements of physical simulations or to influence their flow. In the scene shown on screen, the flames and smoke pass right through the pot because they don't detect it as an obstacle.

To fix this problem, we need to assign a Fluid component of type Effector to the pot, and then select Collision in the dedicated menu.

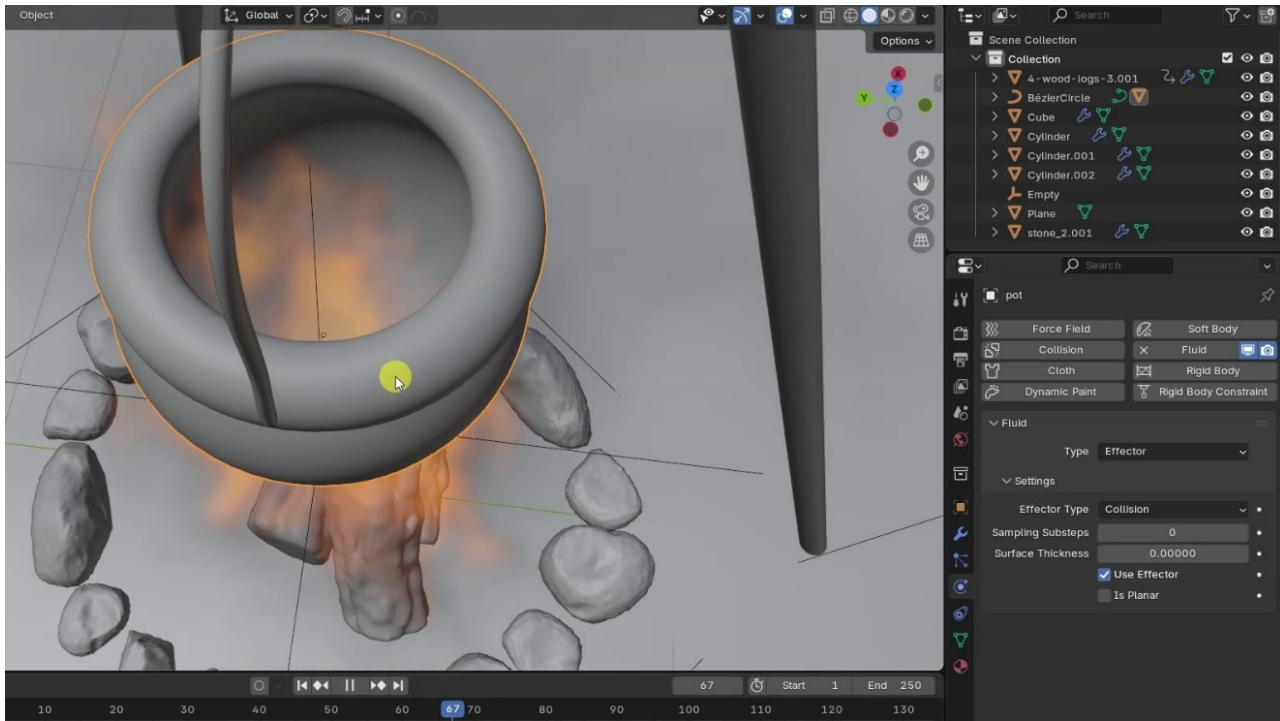


As you can see, there are very few parameters, and in fact most of them we already know from similar ones used with Flow objects. This is the case, for example, with Sampling Substeps, which adds more intermediate calculations between frames.

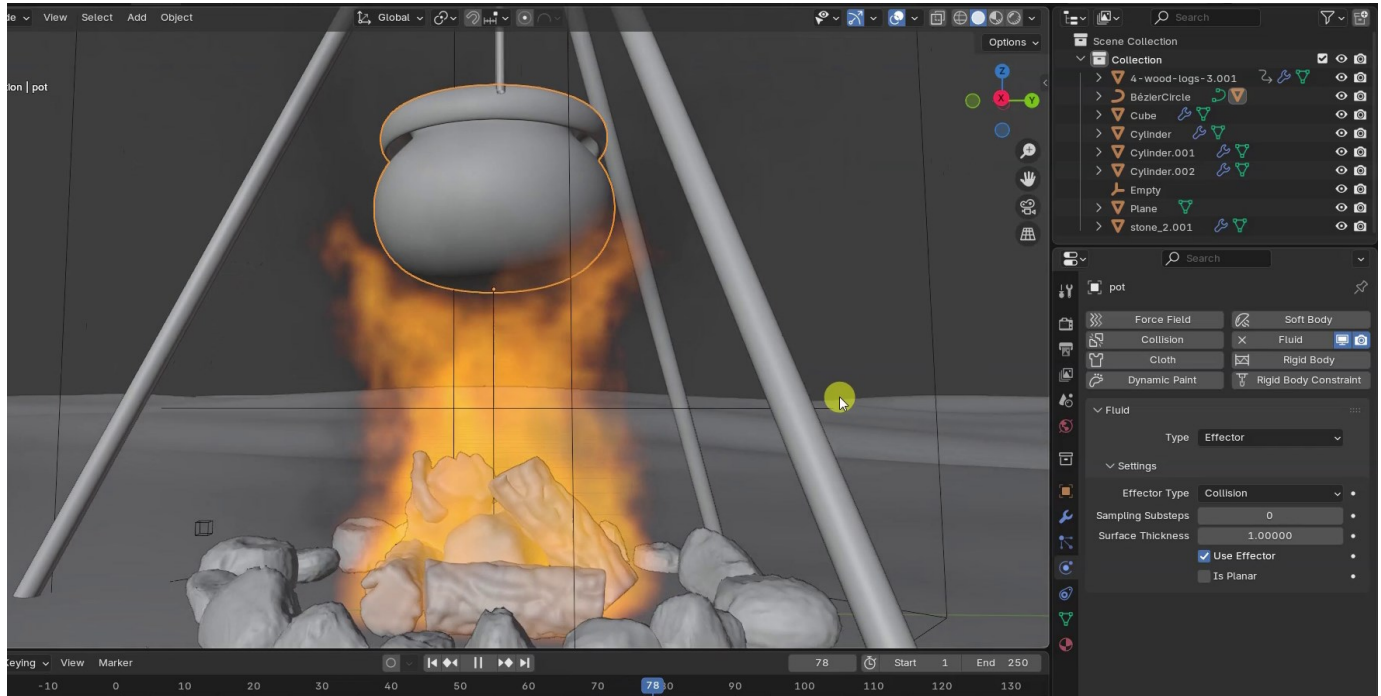
We're also familiar with the Use Effector option, similar to Use Flow, which can be animated and therefore used to enable or disable collisions at will.

We also know that Is Planar is useful for geometries that have no volume or where it's not possible to define an inside and an outside. The new parameter here is Surface Thickness, which creates a kind of outer shell around the object. This can be useful if the simulation causes intersections between fire, smoke, and the object's surface.

Playing the simulation, however, we notice that in some frames the flames still manage to pass through the object.



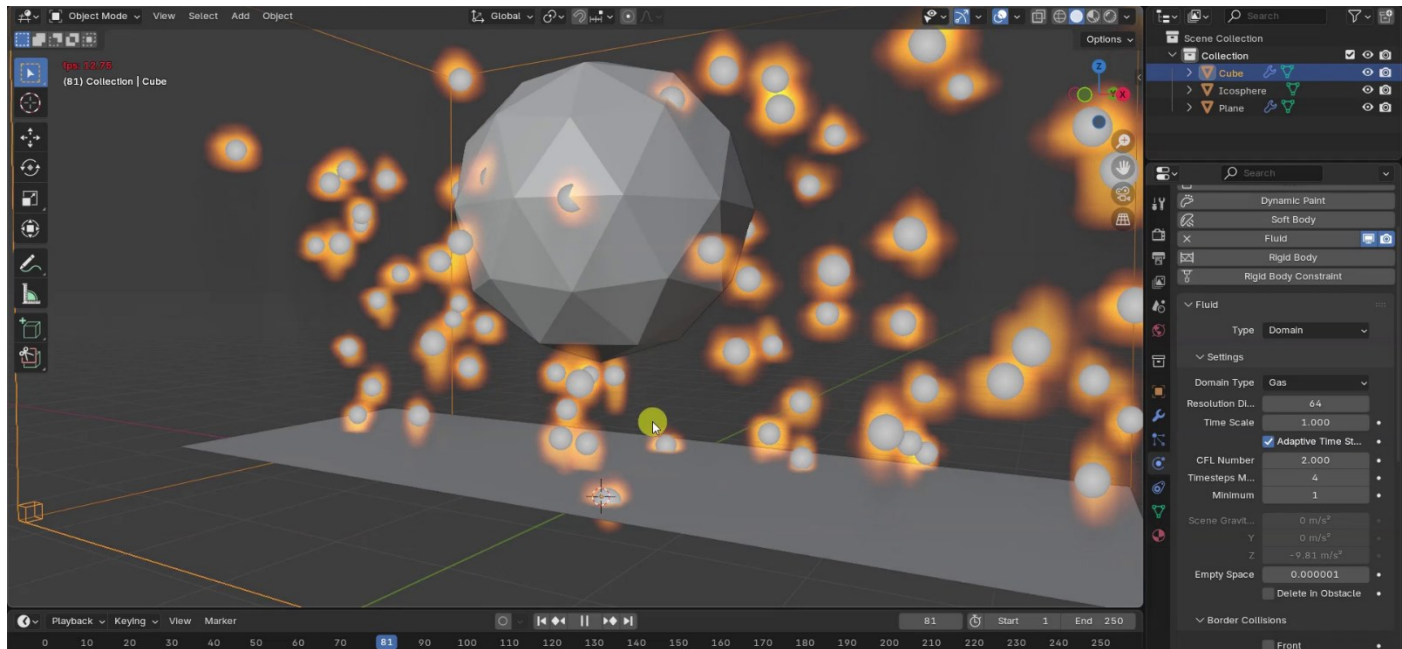
To solve the issue, I kept the same Domain resolution but ran several tests, gradually increasing the Surface Thickness value. With Thickness set to 1, I obtained the result shown on screen. I didn't change the Substeps value, since its tooltip explains that it's mainly useful with fast-moving Effectors. Our object is static, so I only adjusted Thickness, and this is the final result.



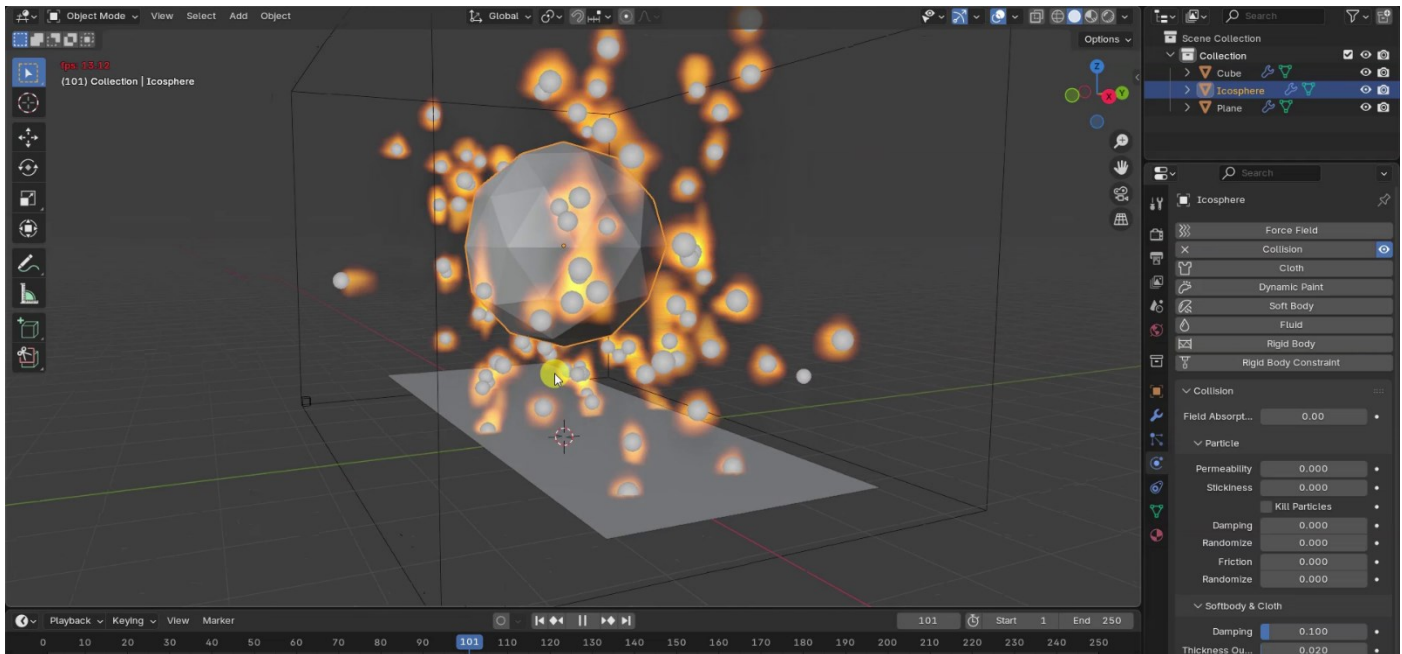
Before moving on to Guide objects, we need to clarify another point about collisions. In the scene I'm using, there's a Plane emitting particles upward. To make the particles rise, I disabled Gravity in the particle system settings. The emitter object is also an Inflow object for the fire simulation. So, I changed the Flow Source to

Particle System and specified the particle system I had created in the corresponding field. To ensure the fire particles aren't much larger and blockier than the system's particles, I increased the Domain resolution.

As you can see, the particles don't interact in any way with the object inside the Domain.



In this case, to make the fire particles bounce when they hit the object, we need to add a Collision component. This type of component makes the particles bounce, but their smoke, and even part of the flames, will still end up inside the object. So, we also need to add a Fluid component of type Collision Effector to properly handle the fire and smoke simulation.



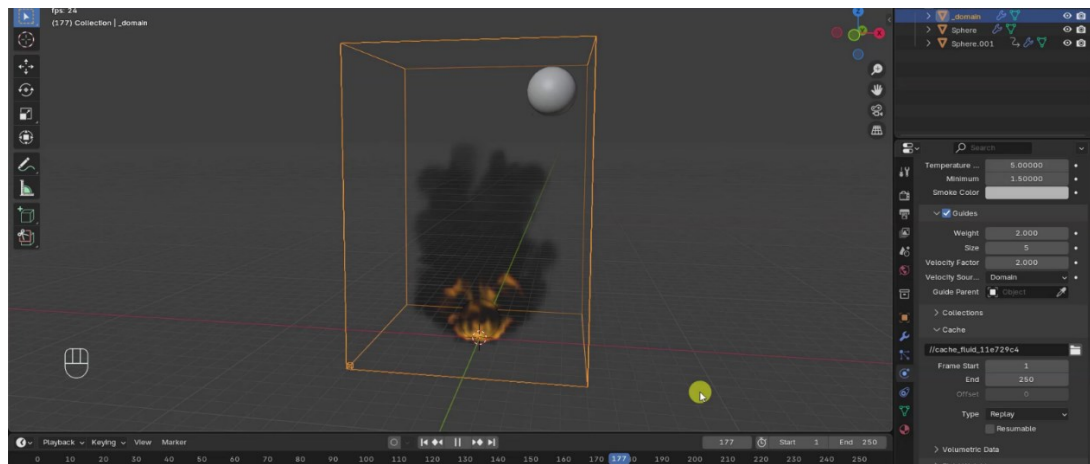
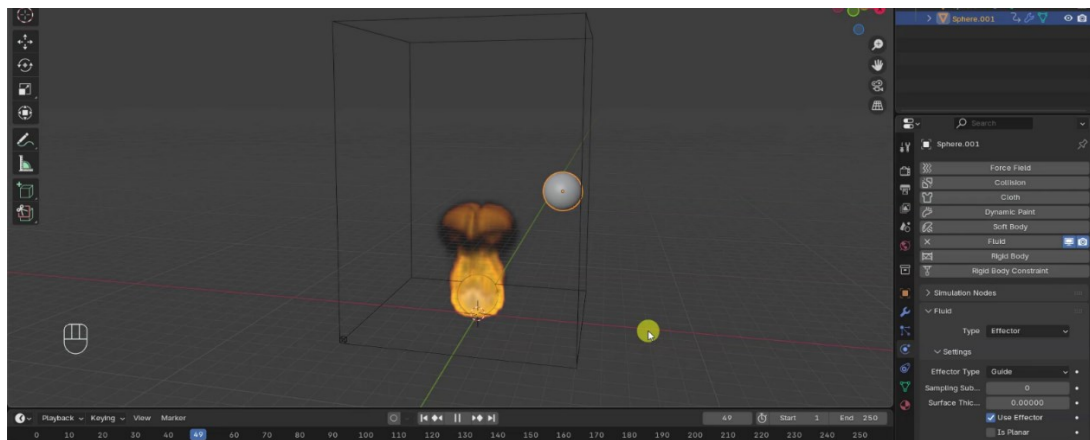
Let's wrap up this episode by talking about Effector objects of type Guides. These are special objects that need to have velocity and that influence the movement of flames and smoke in the simulation. To show you an example, I've added a sphere moving along a strange trajectory inside the simulation Domain. The sphere doesn't yet have any Fluid component, so it has no effect on the simulation.

By adding a Fluid Effector component and choosing Guide as its type, we see some parameters that are already familiar to us, plus a couple of new ones. Among these, Velocity Factor will be the one we're most interested in later.

Playing the simulation, we notice a slowdown in performance but no visible effect of the object on fire or smoke.

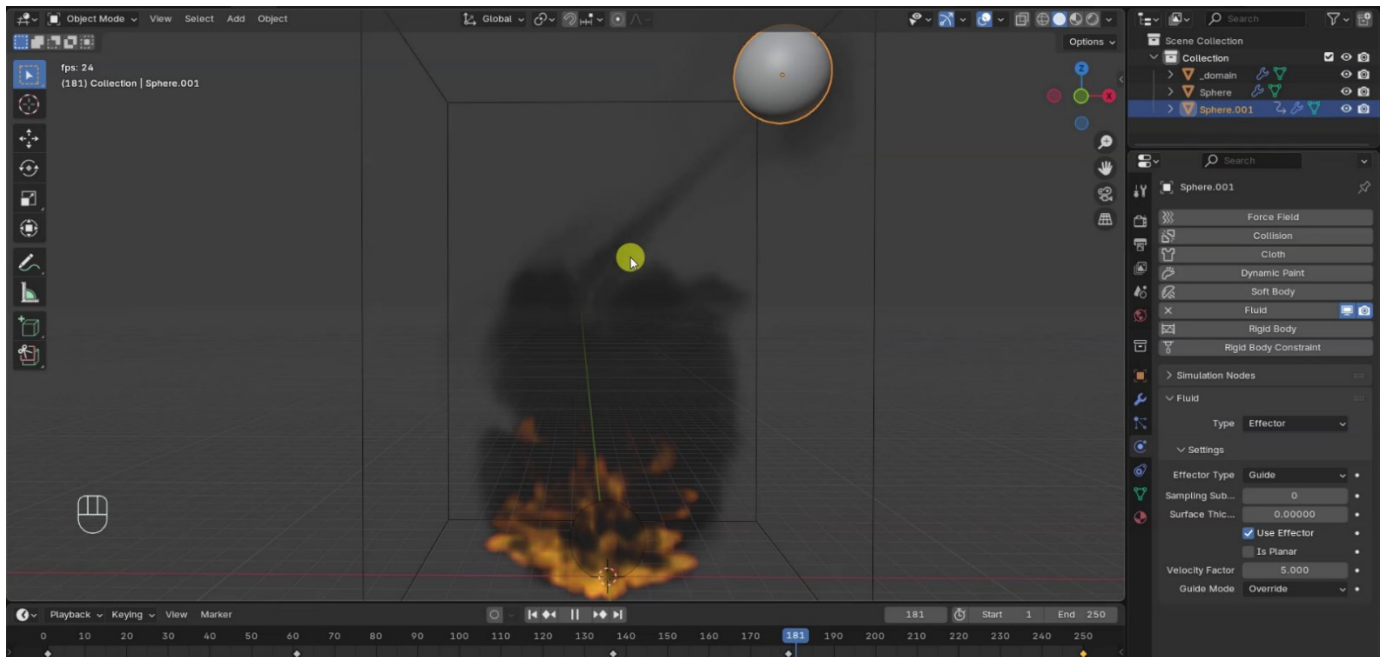
This is because using Guides, which is very computationally expensive, must also be enabled at the Domain level; otherwise, it won't take effect.

After enabling the Guides section of the Domain, I restart the simulation. It now takes much longer to calculate, so I let Blender process it and then show you the result.

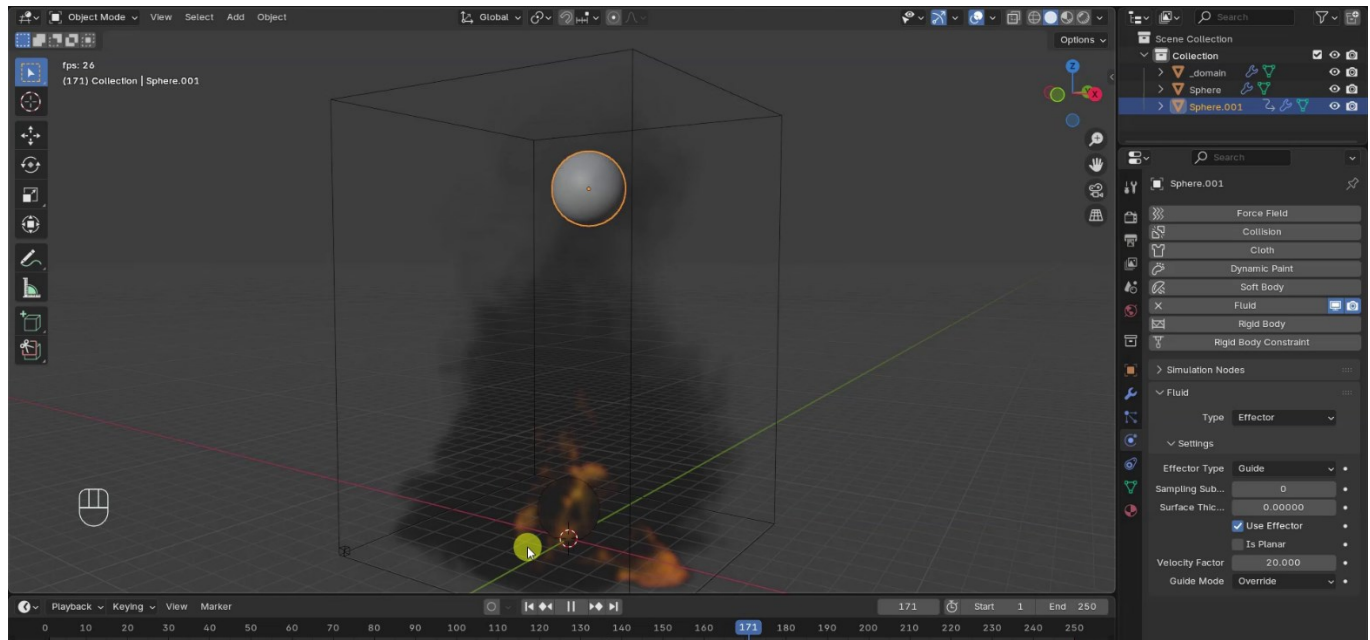


The outcome is already quite different from what we saw without Guides, but it's not very impressive yet. To make the object's influence on the simulation stronger, I select it and test higher values for its Velocity Factor.

This is the simulation with Velocity Factor set to 5. The way the smoke is dragged along becomes more noticeable.



This is the simulation with Velocity Factor set to 20. It's clear that a Guide object is not a Collider, but rather a special type that disturbs and drags the elements of the simulation.



There is, however, another way to influence the movement of fire and smoke, and that's by using Force Fields, which we'll cover in the next tutorial.

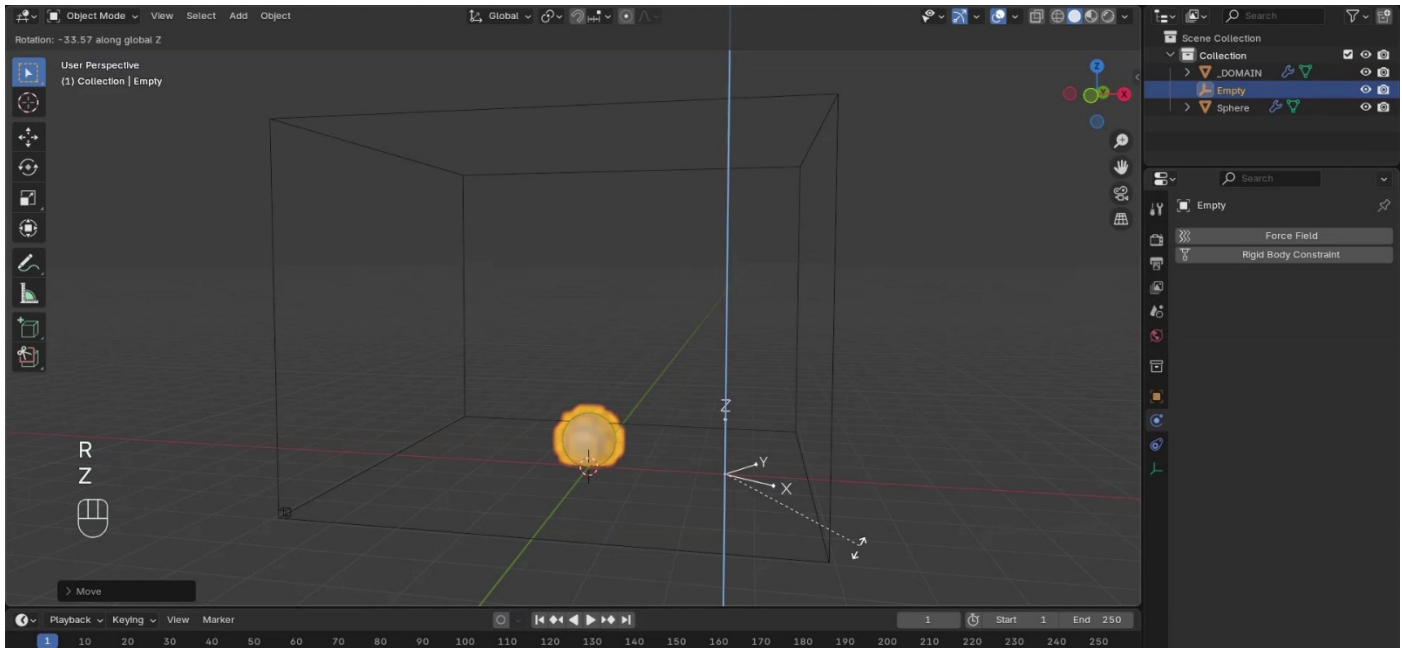
Fire Simulations in Blender 4.5 basics | 9/10: Force Fields (Turbulence, Wind, Curve Guide)

<https://youtu.be/lK5j33JYnBQ>

Hello everyone! In this ninth tutorial of the series on fire and smoke with the Fluid simulator in Blender 4.5, we'll see how to use Force Fields, such as Turbulence and Curve Guide, to influence the behavior and trajectory of the elements in the simulation.

The starting scene I'm using for this tutorial is very simple, consisting only of the Domain and a sphere emitting both fire and smoke in Inflow mode.

By adding an Empty to the scene, we can assign it a Force Field to disturb or guide the simulation elements. Force Fields are found in the Physics panel, just like Fluid simulations for Mesh objects. I chose Arrows as the type of Empty, since some Force Fields act in specific directions, and this Empty type makes them easier to visualize right away.



I then add a Force Field, which by default is of type Force. For each Force Field, a tooltip informs us of its action and the directions in which it operates. Force, by default, has the Point shape and acts radially from its center. However, when playing the simulation, we don't notice any effect, even though the Inflow object seems to be within the Force Field's range.

In these cases, one procedure that seems to work is the following: after saving the project file, specify a cache folder. Then, switch the Domain's Baking mode to All and let Blender calculate the simulation for at least a few frames.

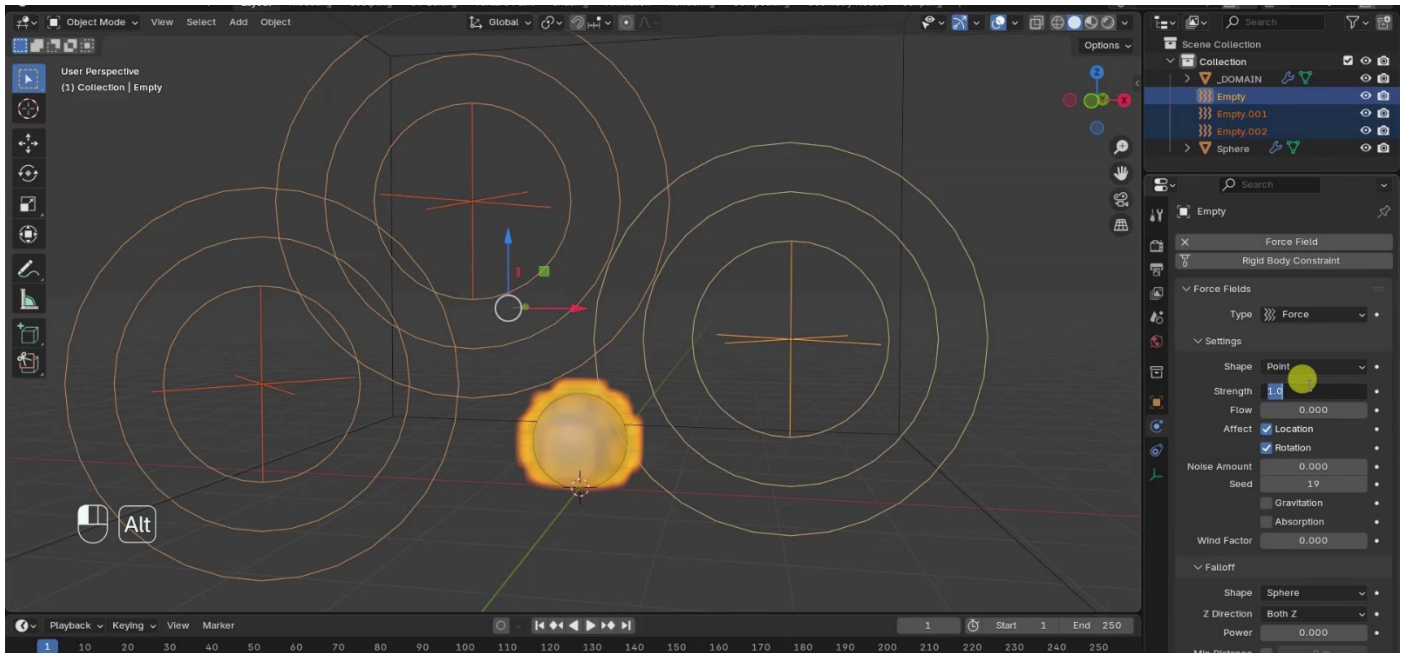
Once this process is done, press Play to see if the simulation has been computed correctly. If it has, clear the Domain Cache and set the Baking mode back to Replay.

Adjust both the project's animation length and the simulation frame range in the Domain, then click Play in the Timeline.

If the simulation now runs correctly, from this point on you can test the different effects in Replay mode.

To control the intensity of the effects, Force Fields provide a Strength field, which applies to the selected Force Field and can be animated.

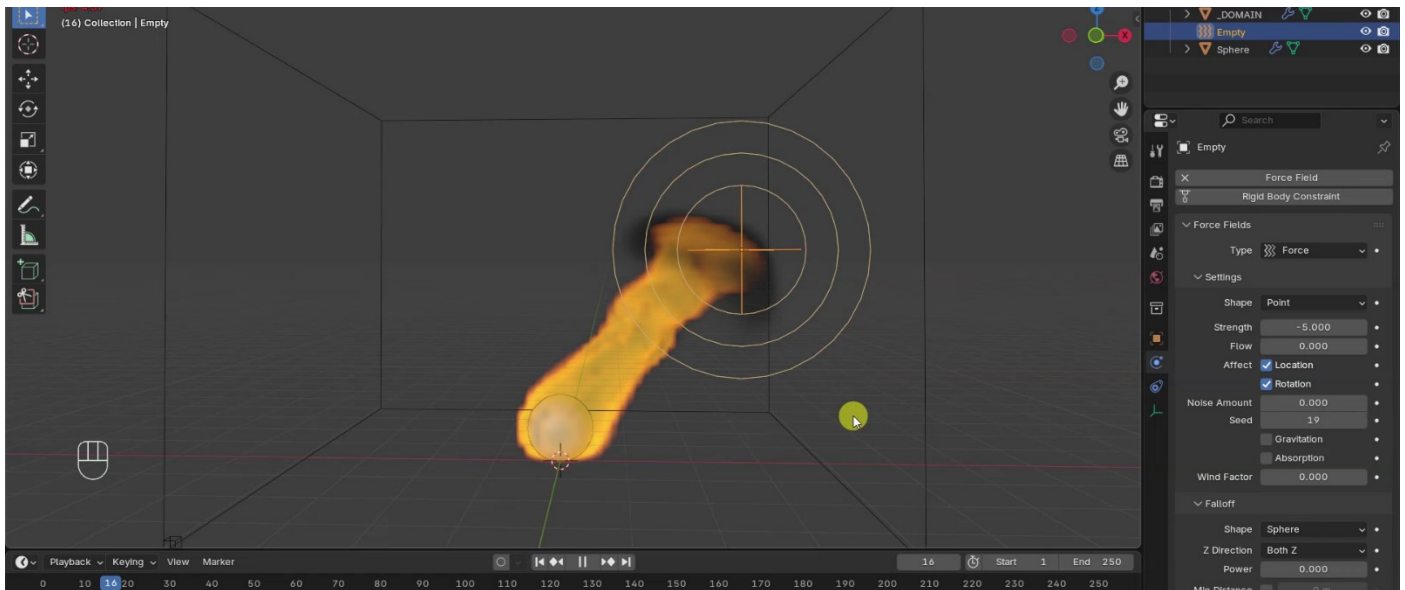
As a side note, you can change the value of a field for multiple selected objects at once if you hold down the ALT key before clicking on the field to edit it.



Strength therefore allows us to adjust, even with animations, the intensity of individual Force Fields. But as we learned in the tutorials on the Domain, it's also possible to control the influence of all Force Fields in the scene, or all Force Fields of a certain type. These settings are found in the Field Weights section of the Domain and, as in many cases, they can also have keyframes, meaning the effect of one or all Force Fields can be animated over time.

The Strength field of Force, as well as many other Force Fields, can also take negative values. In the case of Force, this results in fire being attracted toward the Force Field.

On screen, I'm showing an example created by animating the Empty's movement in the scene and setting Force to a negative value. As you can imagine, this is an interesting way to direct fire and smoke.

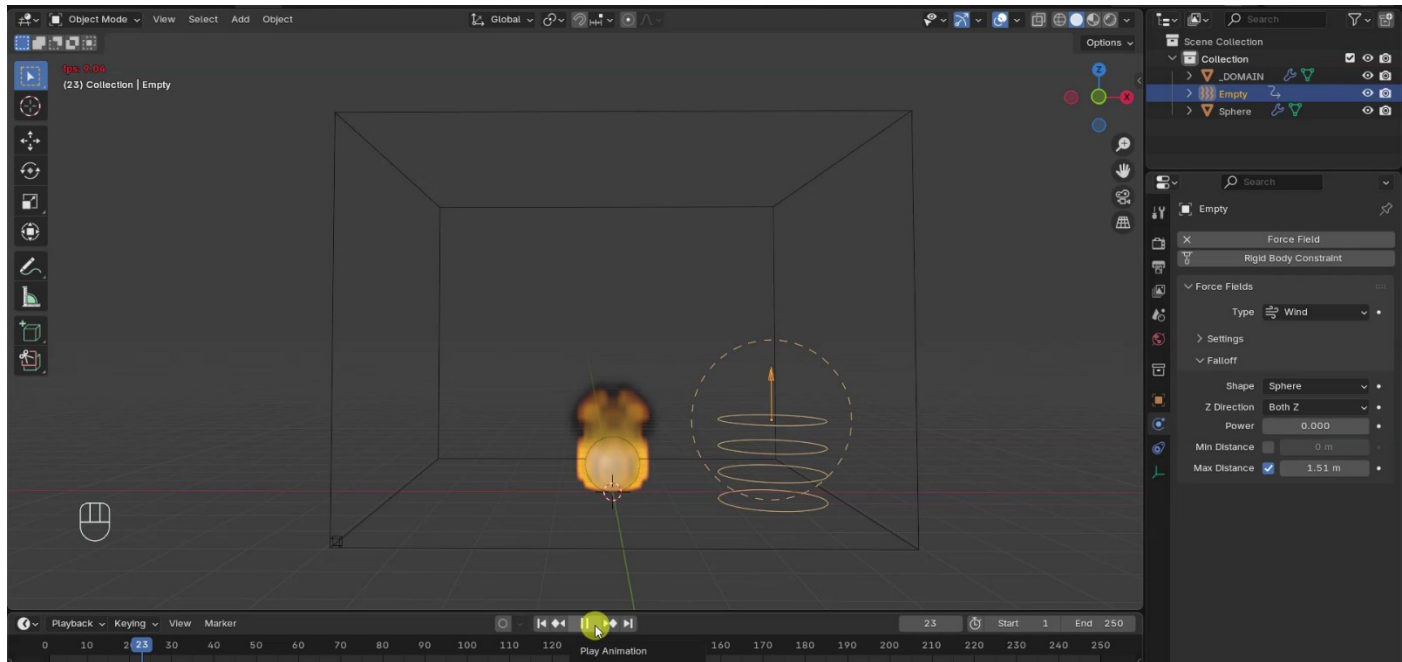


Covering all the parameters of every Force Field available would require a separate mini-series, so I won't be doing that here. However, there's one section worth examining: Falloff, which allows us to set a range for the influence of a Force Field.

At the moment, the Force Field affects the entire virtual space without any attenuation. To show you a clearer example, I'm changing Force to Wind while keeping the Strength negative. Even though Wind acts only on the Z axis and is placed quite far from the Inflow object, the latter is still affected.



To limit the range of influence of the Force Field, we can use the Max Distance field in the Falloff section. By default, this parameter is disabled. When enabled, the Force Field's range is represented by a circle in the 3D Viewport. Like many parameters, Max Distance can also have keyframes, making it another way to control the effects of a Force Field in animations. In this case, Wind is pulling flames and smoke because I left Strength set to a negative value.



I now disable Max Distance, return the Force Field to its original orientation, and set a positive value for Strength to quickly check out a couple of other Force Fields.

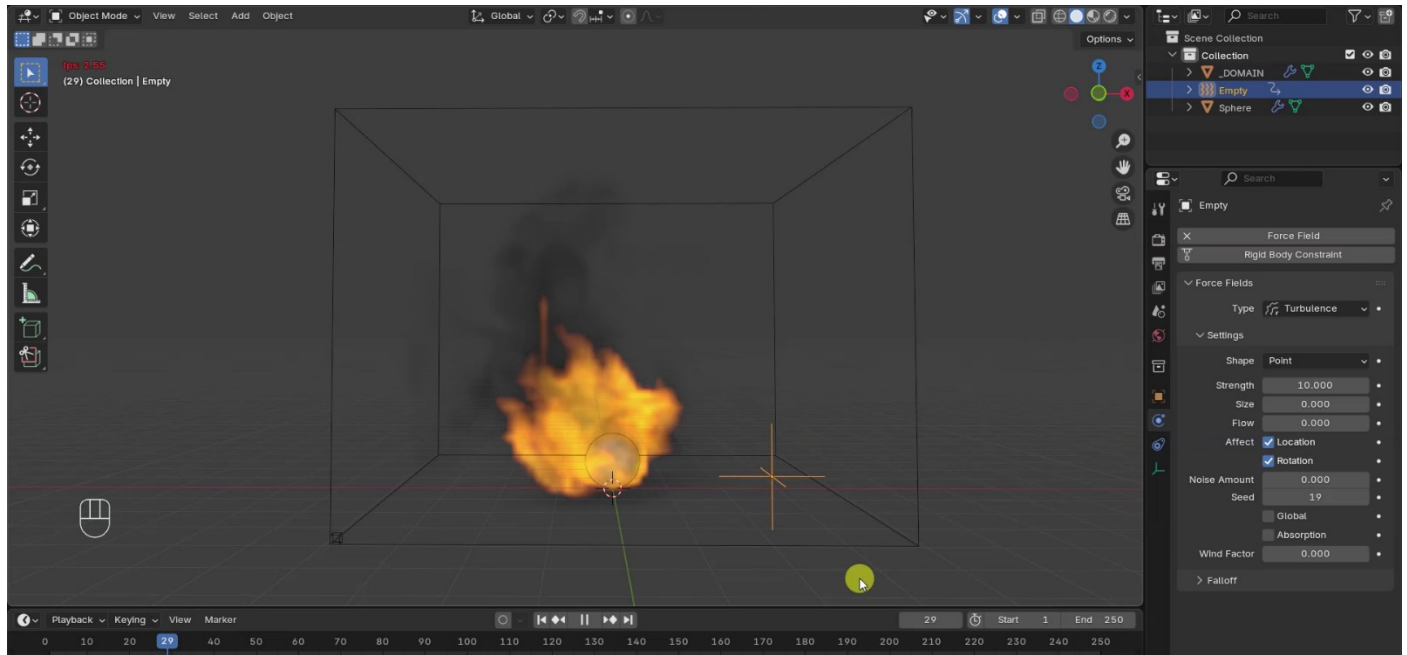
A very useful Force Field for adding some motion to flames is, of course, Turbulence. The disturbance it introduces is so strong that the smoke makes the flames less visible, so I'm reducing it using the Dissolve option in the Domain.

Turbulence can be made even more random by increasing the Noise parameter. This noise is actually pseudorandom, since it follows certain patterns, and you can choose the pattern using the Noise Seed parameter. If your flames look too uniform, even after adjusting Vorticity or adding Textures, a small Turbulence Force Field might be exactly what you need.

To finish, let's take a look at the Curve Guide Force Field, which lets us make simulation elements follow a path. For this example, I recreated the initial scene from scratch, saving the new project in a new file and setting up a new cache folder for the Domain. This way, the initial settings of the Domain and Inflow objects I placed in the scene are the default ones you get when creating them.

However, I'm enabling Dissolve, with a higher frame count than the default, to keep the smoke from obscuring the view too much.

I'm also moving the Inflow object to the side to make room for the path curve.

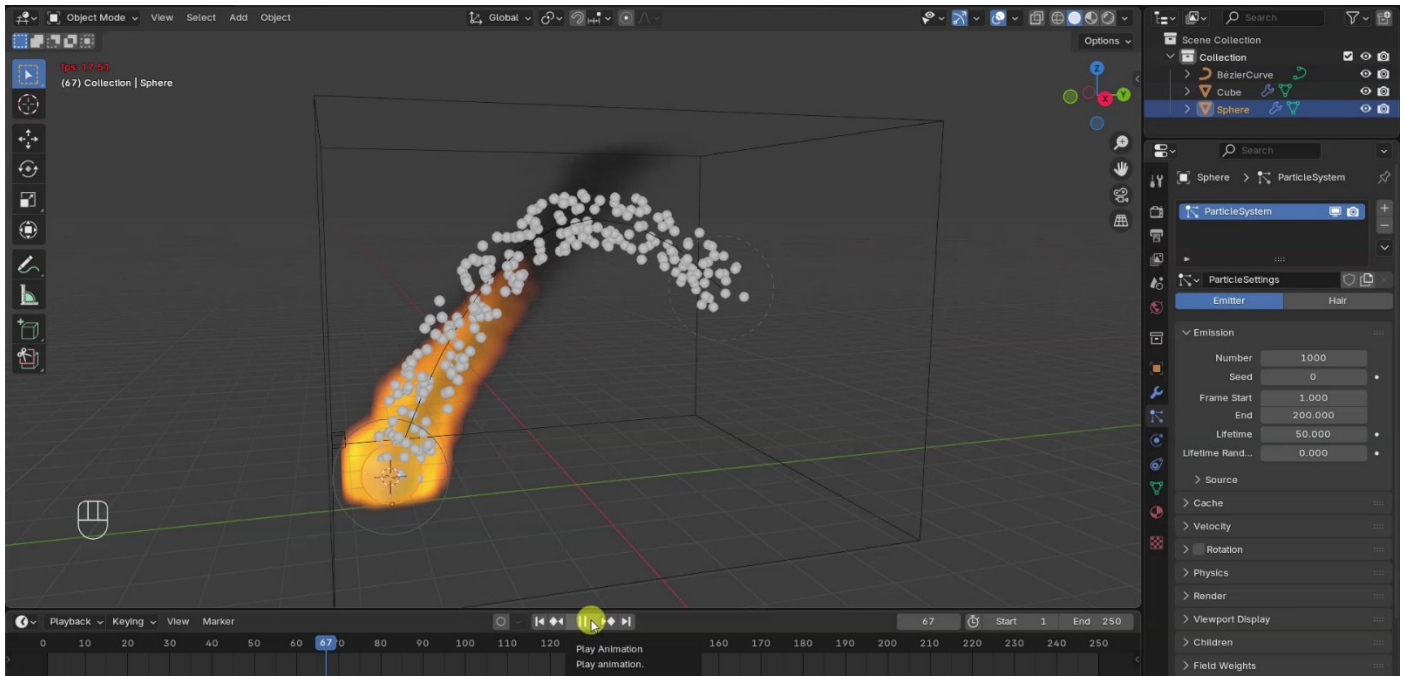


After adding a Bezier curve and positioning it to give it a Curve Guide Force Field, the results are quite disappointing.

If you're familiar with curves and the Curve Guide modifier, you know it's good practice to set the curve's Origin to the start of the path and, most importantly, to make sure the curve has the correct orientation. This can be checked in Edit Mode with Curve Overlays, and an incorrect orientation can be fixed with Switch Direction. Restarting the simulation, things look slightly better, but still not quite right.

I tried various settings, such as disabling gravity in the Domain and increasing the Curve Guide value in the Field Weights. Of course, I also tried lowering Vorticity and Reaction Speed, increasing the resolution, and more. Something did change, but not enough. So let me show you the method I usually use to get the result I want.

I add a Particle System of type Emitter to the object with Inflow and configure the basic parameters so that the particles are emitted properly. Note that the particles immediately follow the curve path, since the emitter is within the Force Field's range.

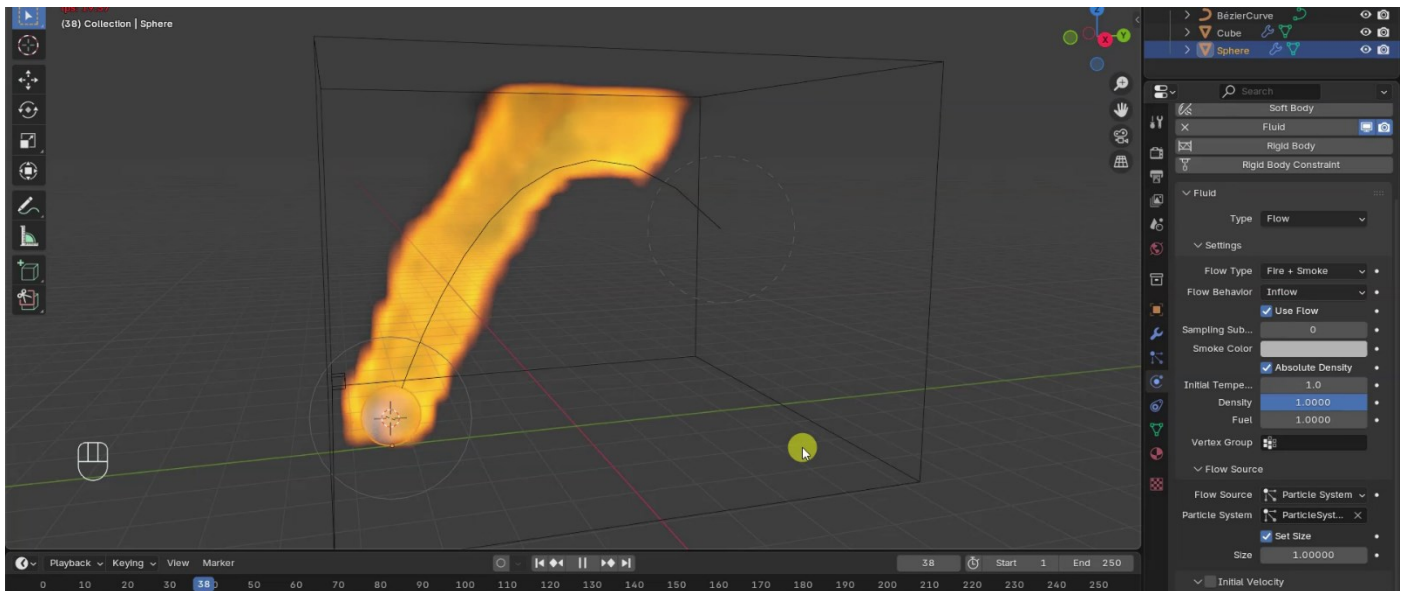


The Force Field's range can be set in its panel using the Minimum Distance parameter. There, we can also set Clumping to 1 so that all the particles converge at the endpoint of the curve.

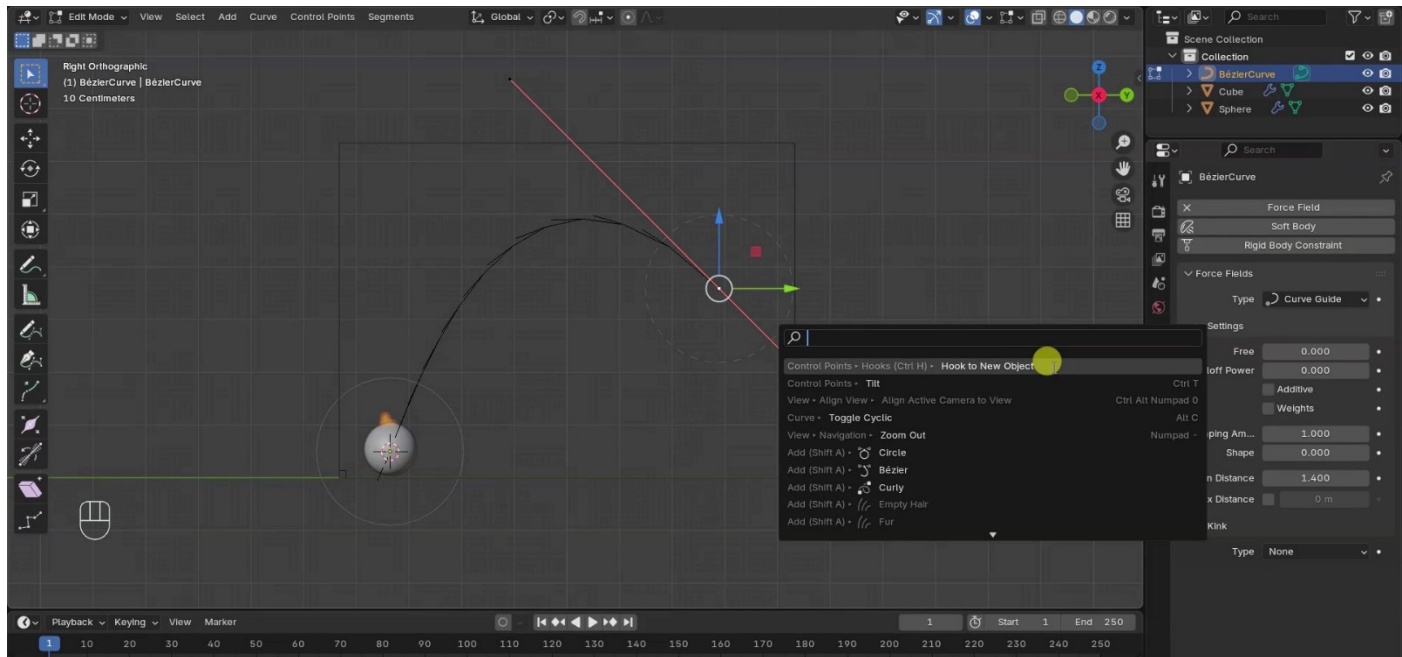
At this point, I set the Render of the emitter's particle system to None, because these particles should not be rendered, and then I go back to the Inflow panel.

In this panel, I change the Flow Source from Mesh to Particle System, specifying the particle system I just created. This way, fire and smoke will be emitted from the particles that follow the path.

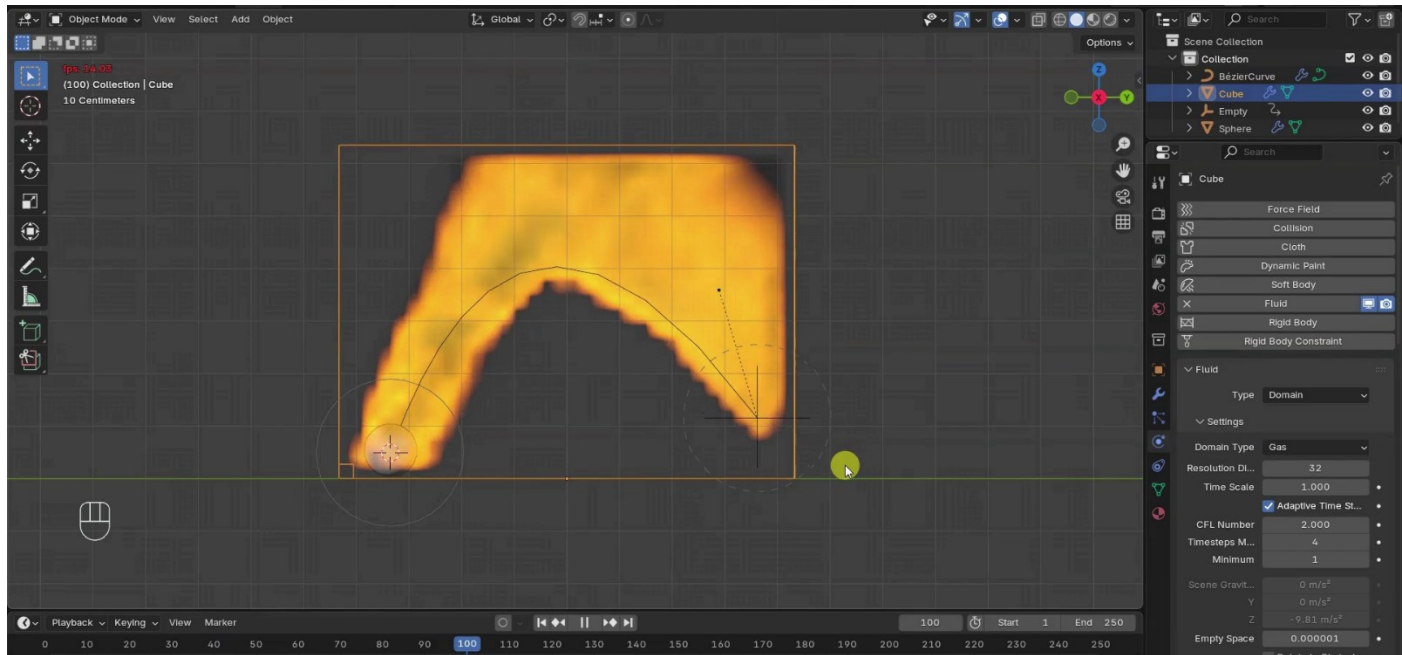
From here on, you can adjust parameters such as Fuel, Vorticity, and others to give the flames the appearance you want.



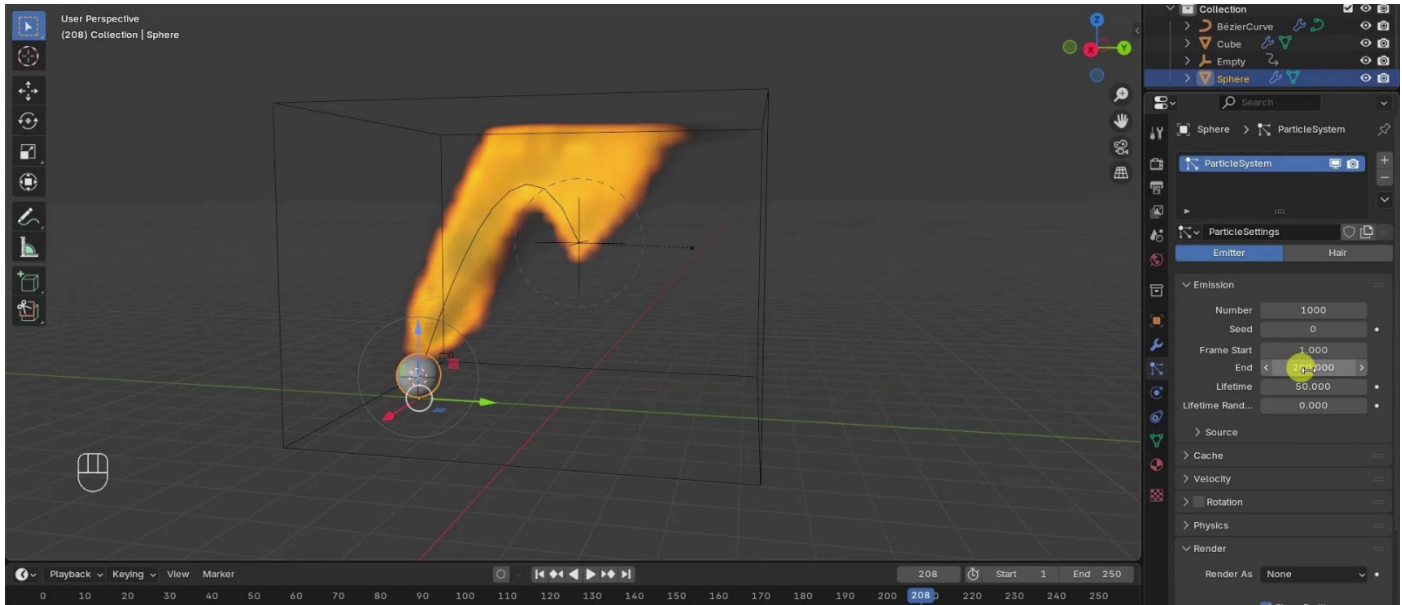
Before ending the tutorial, let me also show you how to link a curve's control point to an external object using a technique called Hooking, which makes it easy to animate the curve's shape. First, I select the curve, go into Edit Mode, and select the last control point. Then I use the Hook operator, choosing the Hook To New Object option, which creates an Empty for this purpose.



In Object Mode, you can move this Empty and animate its movements, deforming the curve and the flames accordingly. Of course, you can use the same technique to add more Empties for other curve control points, creating more complex animations.



As you may have noticed, at some point the particle emission stops. This parameter is called Frame End and is located in the Particle System panel. In the same panel, the Lifetime parameter instead controls how quickly the particles travel along the curve.



Before wrapping up, here are a couple more considerations.

The size of the Emitter object determines the size of the emission surface. So, if you want the particles to be concentrated along a narrower curve, adjust the size of this object.

The number of particles also makes the fire curve look more or less segmented, so you may need to increase it to give the flames a smoother, more continuous look.

On this front, using Motion Blur, which we'll cover in the next tutorial, can also help.

Fire Simulations in Blender 4.5 basics | 10/10: Motion Blur, Filmic, Compositing, Eevee

<https://youtu.be/RDcGQLXRITY>

Hello everyone! In this tenth and final episode of the series on fire and smoke with the Fluid simulator in Blender 4.5, we won't go through full tutorials but rather share tips and suggestions on how to modify the appearance of fire both during rendering and afterward, using Motion Blur, Color Management, and Node Compositing in post-production.

We'll also discuss the settings for rendering fire and smoke in Eevee.

Let's start with a setting that affects rendering itself, without post-production. This is the choice of Color Management, which is made in the Render tab. In particular, the Filmic mode seems to produce flames with more saturated colors. Selecting High Contrast makes the fire appear more intense.

On screen, I'm showing a comparison: on the left, a render made with AgX, and on the right, one made with Filmic and High Contrast. Color Management affects the entire image, not just the fire, as we can see by looking at the stones at the bottom. In a moment, we'll see how to isolate and adjust only the flames in Compositing.



Another option we can enable in the Render tab is Motion Blur, which is very useful when rendering these simulations since flames are never perfectly still between frames. In a previous episode, we mentioned that the Domain has a field related to Velocity Scale for Motion Blurring, but Motion Blur must also be enabled at the rendering level in the Render tab. In this tab, the Shutter parameter defines how many frames before and after the current one are taken into account.

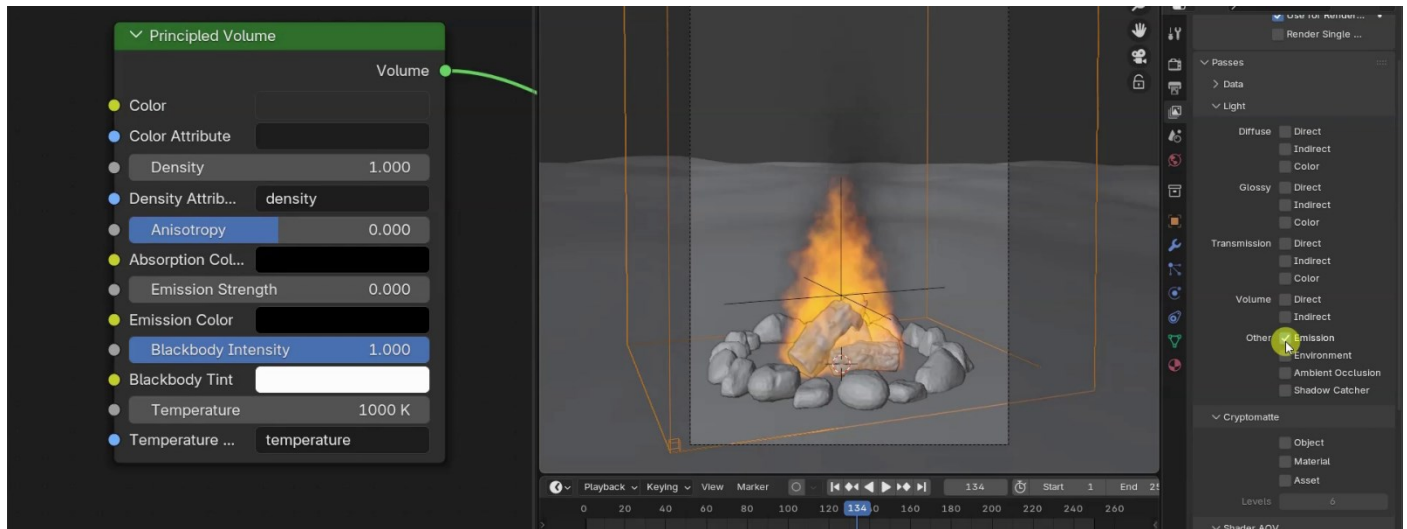
Higher values result in stronger motion blur.

On screen, I'm showing a comparison between a render with Motion Blur disabled, one with Motion Blur enabled and the Domain's Velocity Scale set to 1, and a render with Motion Blur enabled and the Domain's Velocity Scale set to 10, intentionally exaggerated to highlight the differences. Unfortunately, there's no magic formula for finding the right value for Shutter or Velocity Scale, since the best choice depends on the needs of your current project. Testing is necessary. Keep in mind that enabling Motion Blur and increasing Shutter or Velocity Scale values will significantly increase rendering times.



For any rendering, the final look also depends on good Node Compositing in post-production.

Fire is no exception, and in this case, we can also take advantage of the Emission Render Pass, which allows us to isolate the pixels of light-emitting sources, such as Emission or Volumetric Materials, for compositing. So, I enable this Render Pass in its dedicated panel before running a test render and moving to the Compositing Nodes editor.



Using the Emission information, which gives us the flames in color on a black background, we can apply various Compositing nodes to improve blur, glow, contrast, or even color.

The result of these operations can then be applied back onto the original render. Blender offers different blending modes and factors for this purpose.

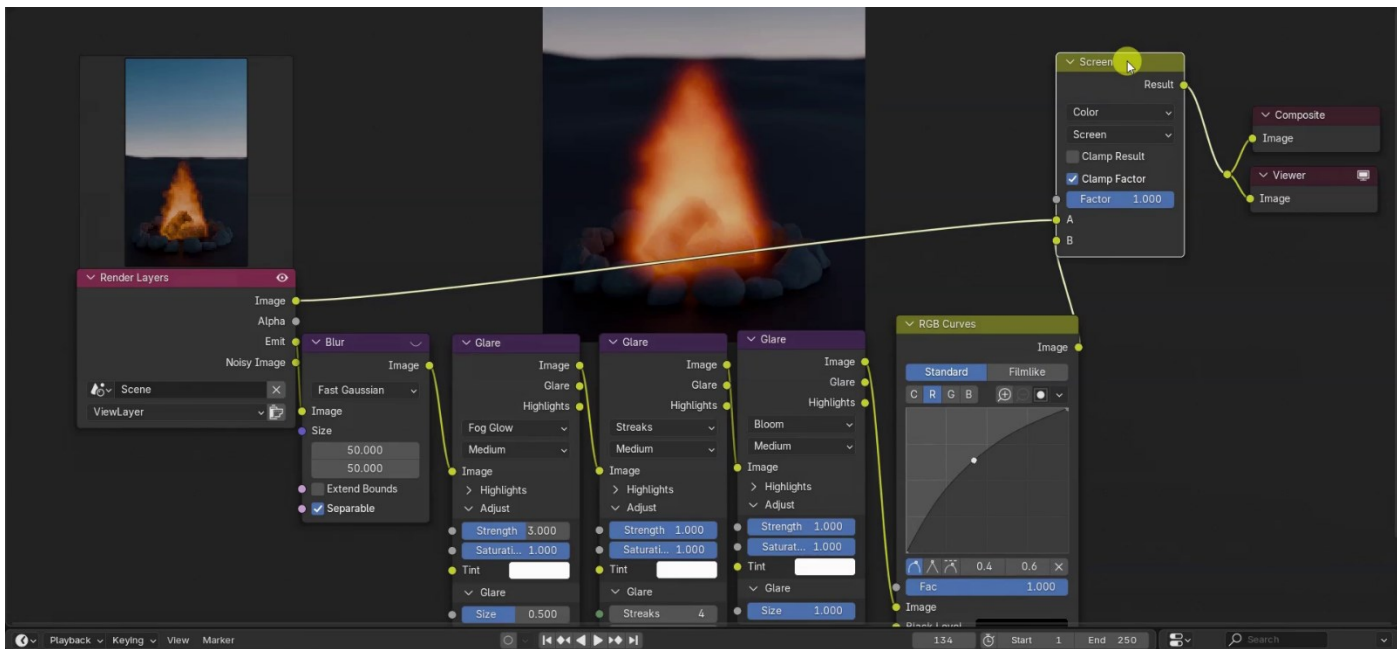
For example, I've processed the flames separately, blurring them and adding different types of Glare effects, before applying the result to the original render through a Mix Color node set to Screen.

I'm also quickly testing brightness adjustments using the C channel of an RGB Curves node.

I even try modifying only the Red channel within the same node.

The final result may be a bit too strong, but I can always adjust the Factor value as needed.

I can also disable or re-enable the entire setup by selecting the Mix Color node and pressing M, that is the Mute function, which toggles the node on or off.



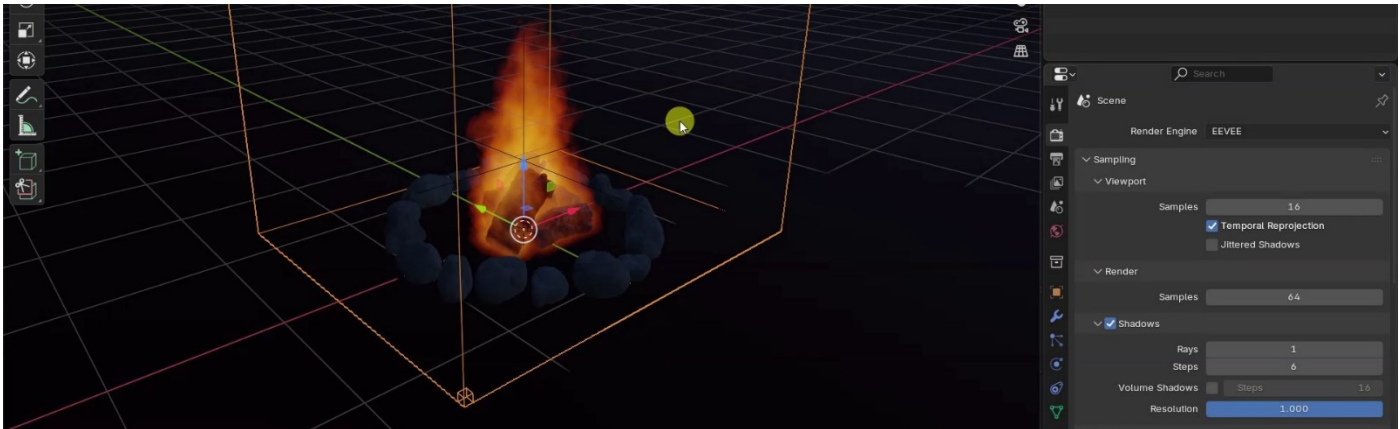
Before moving on to the tips for rendering settings in Eevee, let me show you a final comparison of the options we've discussed so far. On the left, you can see a render with AgX as the Color Management, without Motion Blur and without Compositing. On the right, you can see the same scene rendered with Filmic set to High Contrast, Motion Blur with Shutter set to 1 and Velocity Scale set to 10, plus a bit of post-production done with the node setup I just described.



OK, now we can move on to the rendering settings in Eevee.

Up to now, I've always used Cycles for rendering, but Eevee introduces some necessary differences since its key feature is real-time rendering.

In Eevee, fire and smoke simulations are represented as slices of the scene that are then stacked like translucent sheets. Among the parameters to configure are the start and end points of these slices relative to the observer's position. The settings that control the quality of the final result are mainly found in the Render tab, specifically in the Volumes section.



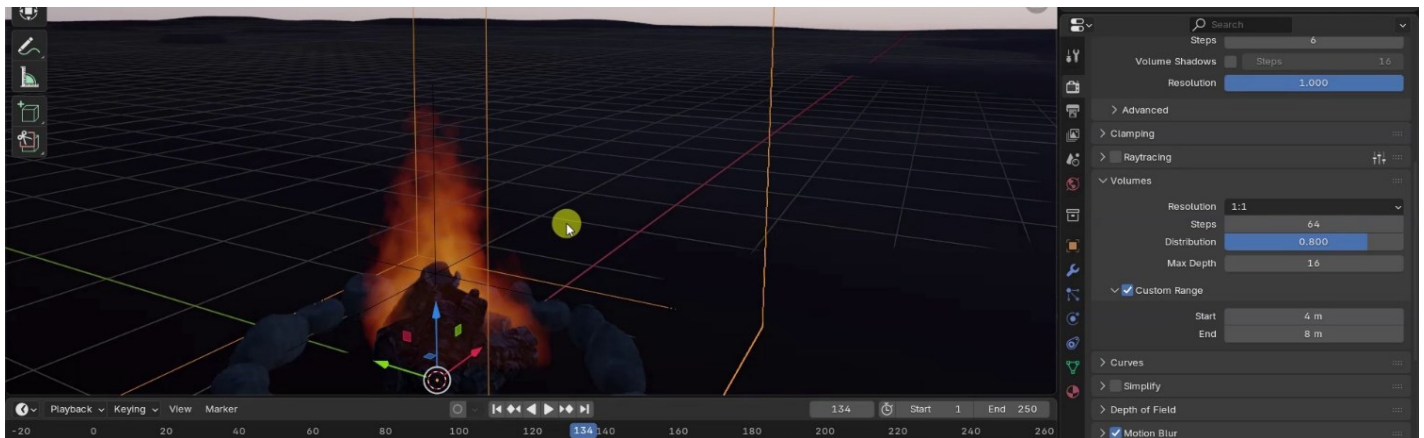
The first parameter in this section is Resolution, and intuitively, the closer this value is to a one-to-one ratio, the higher the quality will be. With lower ratios, quality decreases, but render times are shorter.

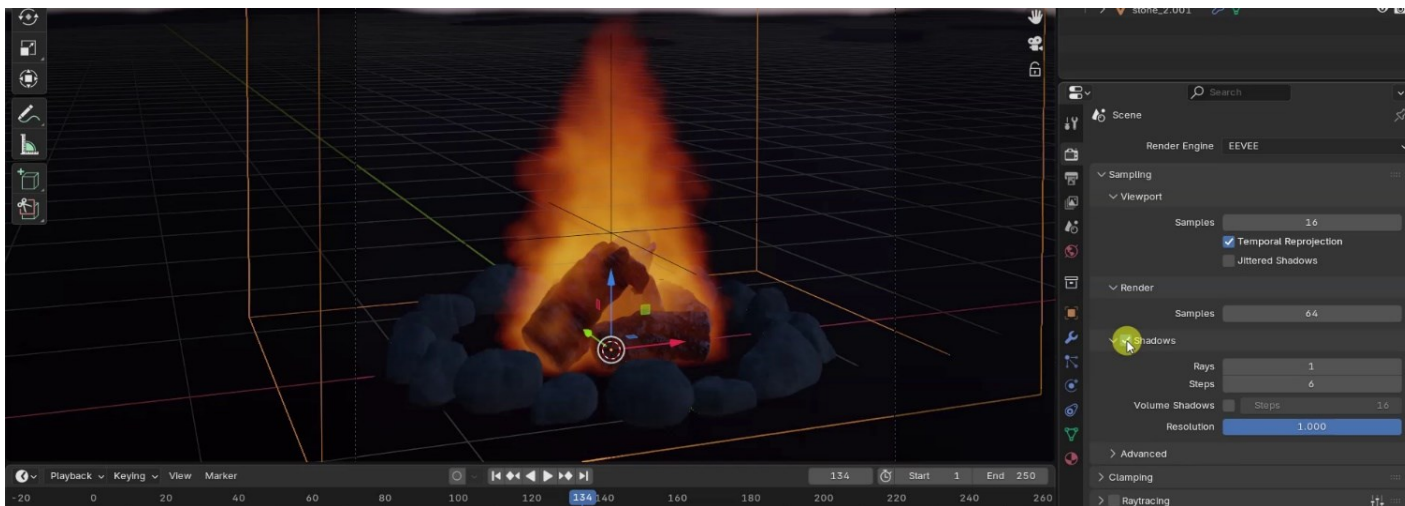
As mentioned earlier, the viewing volume is divided into slices called Steps. The Steps parameter works together with the Custom Range Start and End options, which are enabled by default. The space between Start and End, beginning from the observer's position forward, is divided into the number of

slices specified by Steps. Each slice is rendered, and the combined result produces the final image. Obviously, we don't need the default 100 meters, so we can reduce the Start and End values to match the Domain of the simulation and concentrate the Steps within that space to achieve a better result.

These are the most important parameters to understand in this section, especially to explain why simulations sometimes disappear when moving around in the scene!

As for shadows and their quality, in the Sampling Shadows section of the Render tab you'll find the Volume Shadows checkbox, which is disabled by default. Enabling it allows you to also set the Steps value, which, of course, is tied to the quality of the result.





And that wraps up this final tutorial in the series on the basics of Fluid simulation for fire and smoke in Blender 4.5! I hope this series has been useful to you.

If you'd like to thank me, the best way is to subscribe to the channel and leave a Like on the video.

Thanks, and see you soon!